

**PRODUCT MODEL-BASED ANALYTICAL MODELS (PBAMs):
A NEW REPRESENTATION OF ENGINEERING ANALYSIS MODELS**

A Thesis
Presented to
The Academic Faculty

by

Russell Speights Peak

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in Mechanical Engineering


Georgia Institute of Technology
November 1993


Copyright © 1993 by Russell Speights Peak

**PRODUCT MODEL-BASED ANALYTICAL MODELS (PBAMS):
A NEW REPRESENTATION OF ENGINEERING ANALYSIS MODELS**


Approved:

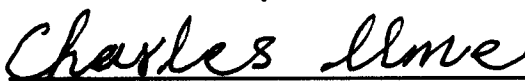

Robert E. Fulton, Chairman


Dereje Agonafer


Ronald C. Arkin


Jonathan S. Colton


James I. Craig


I. Charles Ume

Date Approved 11/22/93

DEDICATION

To Haruko, Joy, and Jonathan

PREFACE

In a sense, this thesis began several years ago when I was working as a product designer at AT&T Bell Laboratories in Middletown, New Jersey. I needed to design a cantilever spring with non-uniform cross-section as part of an injection molded plastic piece for a business telephone. After searching through several textbooks, handbooks, and design notes, it struck me how ill-prepared I was to apply what I had learned in my undergraduate mechanics of materials course (in which I received an A) to this relatively simple "real world" design situation. The problem had not been neatly setup for me, and there were not any well-defined questions, much less clear-cut answers in the back of the book. Sifting through the detailed design information of this plastic part (all its dimensions, tolerances, fillets, surface specs, material data sheets, etc.) to try and create a simple analysis model was frustrating and humbling, yet, at the same time, enlightening and intriguing. This thesis deals with the challenging intersection of engineering design and engineering analysis which had confronted me for the first time in the form of that plastic part.

The overriding impression I have as I look back on this thesis is that I have tried to do too much. It has definitely been at least three or four handfuls¹, and has come with much toil and chasing after the wind. As such, some aspects are not as strong as others, and consequently, are good candidates for future extensions. In particular, Chapter 7 (Development Guidelines), Chapter 8 (Implementation Guidelines) and the operations section in Chapter 5 include the word "preliminary" in their titles to indicate this situation. Still, I feel that the overall thesis is aided by their presence, at least to the extent of giving the full perspective of the PBAM representation. Also, future experiences to be gained by representing a wider sampling of analysis models as PBAMs undoubtedly will help refine this work. In spite of these limitations, I hope the reader finds this thesis to be at least a helpful step towards achieving advanced design and analysis integration.

¹ See the quote for Chapter 4.

ACKNOWLEDGMENTS

*Iron sharpens iron,
So one man sharpens another.*
Proverbs 27:17 [NASB]

Undoubtedly many organizations and individuals have contributed to this thesis both directly and indirectly. First, let me gratefully acknowledge the financial assistance I have received as a Georgia Tech Presidential Fellow and as a Graduate Research Assistant (GRA) these past four years. My work as a GRA, briefly sponsored by the School of Mechanical Engineering, primarily has been supported by the Georgia Tech Manufacturing Research Center (MARC) and its industrial sponsors: DEC, Ford, IBM, Motorola, and the U. S. Army Missile Command.

Special thanks goes to John Maloney for supplying the updated version of ThingLab II [Maloney, 1991] that I used in the case studies.

The members of my thesis committee have been most helpful and patient with me. Though it has been painful and has required many hours of labor, I know that the reorganization and refinements that they requested at my defense in August has made this a better thesis.

Dereje Agonafer of IBM-Poughkeepsie has been an encouragement from the industrial side as his comments have reaffirmed the need for this type of integration between design and analysis. As the co-chairman of the electronic packaging CAD session at the ASME Winter Annual Meeting for the past several years, he also has been helpful and patient in handling our submissions to that session.

Ron Arkin, College of Computing, introduced me to artificial intelligence in his applications-oriented course a few years ago. He has managed to keep us engineers straight when it comes to computer science terminology and concepts, and has pointed me in the right computer science direction numerous times in this thesis.

Jon Colton, Mechanical Engineering, has lived up to his reputation as being one of the toughest committee members - and its because of this that I would recommend to others to have him on their committee, too, if they are serious about having a good thesis. I now appreciate his thoroughness and insightful comments on earlier drafts of this thesis.

I owe him a special debt, too, for generously allowing me to do most of my implementation work on his machines in the Laboratory for Intelligent Design Systems (LIDS).

The fact that inverse problems and multidirectional I/O are considered in this thesis is due in large part to Jim Craig from Aerospace Engineering. His analysis experience in a different discipline and his hard questions about model interaction have all contributed to this thesis.

Charles Ume, Mechanical Engineering, has been a big encouragement to me and has helped me with many everyday matters at the Advanced Electronic Packaging Laboratory (AEPL). It has been a pleasure to work for him and Bob Fulton in this lab which I got to see established and grow during my stay at Georgia Tech. I really appreciate his patience with me as I have hogged the AEPL computers, scattered my work throughout the lab, and spilled coffee here and there in the production of this thesis.

I feel very privileged to have had Bob Fulton as my "Doktor Vater" and the chairman of my thesis committee. He has taught me much about the business of doing research - from doing proposals, to giving presentations, to writing reports, to attending conferences and visiting sponsors. I particularly appreciate the importance he places on publicizing research work in person, to the point that I have traveled almost as much as a graduate student as I did as an employee at AT&T Bell Labs. His longer term view of how the interactions experienced in such travel benefit a research project is unusual, yet, I believe, exactly correct.

His guidance in the big picture of the research and his past engineering analysis experiences have been invaluable. I am thankful for his enduring my meticulous ways and, even more, for helping me see other ways to go and escape them (sometimes). Furthermore, I am grateful to him for how he has always treated me with respect as a junior research colleague rather than as a lowly graduate student.

Time and space do not permit me to acknowledge others as I would like to; instead, I simply will list a few names and organizations (both present and past members) here who have helped me and my family in this work:

- Doktor Brüder und Schwestern: Angela Burkes, Bipin Chadha, Selçuk Cimentay*, Tal Cohen, Chien Hsiung, Gintas Jazbutis, Deeptendu Majumder, Ravi Rangan, Andy Scholand*, Srivatsa Shamanna*, Philip Su, Sang Synn, Diego Tamburini*, Ching-yang Wang, Chao-pin Yeh*, Wen Zhou*.
- AEPL Lab: Michael Gabertan, Jeff Garratt, Chiayu Fu, Jiandong Mao, Tim Martin, Mike Stiteler, and Frankie Tsang. Those marked above with an asterisk (*) also fall into this category. Several of these people have helped review parts of this thesis, for which I am grateful.
- LIDS Lab: Patsy Brackin, Susan Carlson, Lisa Dixon, Paul Lomangino, Judd Staples, and Paul Vallis.
- Design Group Meetings: Janet Allen, Bert Bras, Jon Colton, Farrokh Mistree, Dave Rosen, plus their armies of students.
- Other GT Associates: Bonnie Diaz, Vincent Fox, Norma Frank, George Graham, Mike Ingram, Claudette Noel, Eric Stephens, Bill Wepfer, Melinda Wilson and numerous others.
- AT&T Colleagues: Tinyee Jue, Rex Loyer, Usul Suljoadikusumo, Paul Yuan, and many others.
- Atlanta First Baptist Church International Department, including Pierre Ayereby, Naresh and Veena Malhotra, Alan Yang, and many others; New Monmouth Baptist Church (NJ); and Norcross Baptist Japanese Mission.
- Other Special Friends: Mark and Jeanine Coleman, William Fondo, Kevin and Shawn Frost, Kevin and Bonnie Gue, Richard Hipp, Dwight and Miriam Singer, Ching-yang and Tsu-li Wang, David and Jan Wilkensen, and Chao-pin and Winnie Yeh.

I am thankful for my family, their encouragement and prayers, and the constant support that they have been through these past four years. Besides my wife and children, my immediate family members are Masao Kinoshita, Naoko Kinoshita, Shin'ichi and Michiko Kinoshita, Ed, Reiko, Shila, Christine, and Andy Taylor, Floyd and Chichi Peak, Louis Peak, and Marianne Peak.

Words are inadequate to express what my wife and two children have meant to me throughout our time at Georgia Tech. This thesis is truly theirs as well because of all the work, sacrifice, encouragement, and proper perspective that they have put into it. Thank you, Haruko, for believing that I could make it, and for enduring all of my slowness. Without your help in word processing and figure creation, and your wise counsel, I would still be in the MARC Building now, probably changing fonts and adjusting spacing until everything looked absolutely perfect. I'm sure all your home baked goodies for the thesis committee meetings did not hurt either. Joy, my sweet little girl, thank you for your kind notes and drawings that have encouraged me during many difficult moments. Jonathan, my little buddy, your early rising and consistent desire to play with me each morning has helped me remember what is really important.

Finally, I would like to thank Him who has been my greatest encouragement throughout the past four years - my Creator, my Redeemer, my Helper, and my Friend. By His grace, He has guided, helped, corrected, exalted, humbled, protected, and exhorted me. His Word, especially Psalms and Proverbs, has been a continual source of comfort, guidance, and wisdom. Also, reading how Joseph, David, and Daniel, by God's grace and favor, honored Him in their "secular" work² has been particularly motivating and inspiring. As the familiar song goes, so has been my experience:

God is so good, He's so good to me.

*Blessed be the Lord, my rock, who trains my hands for war, and my fingers for battle;
My lovingkindness and my fortress, my stronghold and my deliverer;
My shield and He in whom I take refuge.*

Psalm 144:1-2 [NASB]

*How blessed is he whose help is the God of Jacob,
Whose hope is in the Lord his God;
Who made heaven and earth, the sea and all that is in them.*

Psalm 146:5-6 [NASB]

² See, for example, Genesis 39:21-23, II Samuel 5:10,12, and Daniel 1:9, 6:3,25-28.

TABLE OF CONTENTS

PART I PROBLEM DEFINITION

CHAPTER 1 INTRODUCTION	2
CHAPTER 2 PROBLEM DESCRIPTION.....	8
2.1 Introductory Definitions	8
2.2 The Engineering Analysis Process	13
2.3 Automating Routine Analysis - Process Viewpoint.....	19
2.4 Automated Routine Analysis - Integration Viewpoint.....	22
2.5 Summary.....	25
CHAPTER 3 RELATED WORK.....	26
3.1 Automated Modeling in Engineering Analysis.....	26
3.2 Analysis Model Generation Using Product Model Data	31
3.3 Representation of Analysis Problems	31
3.4 Representation of Analysis Models.....	32
3.5 Summary of Gaps.....	34
CHAPTER 4 OBJECTIVES FOR ANALYSIS MODEL REPRESENTATIONS	35
4.1 Thesis Objectives	37
4.2 Other Objectives	54
4.3 Summary.....	65

PART II ANALYSIS MODEL REPRESENTATIONS

CHAPTER 5 THE ANALYTICAL BUILDING BLOCK REPRESENTATION	67
5.1 Overview of Constraints.....	67
5.2 Overview of Analysis Model Object Representations	76
5.3 Constraint Schematics of Analysis Models.....	78
Example 5.1 Analytical Primitive (Elementary Rod).....	82
5.4 Object Relationship Diagrams	85
5.5 Subsystems	88
Example 5.2 Multibody Analysis Model (Interconnected Rods System)	92
5.6 Analysis Model Options	94
5.7 Extended Constraint Graphs.....	100
Example 5.3 ABB with a Subsystem (Safe Rod).....	101
5.8 I/O Tables	103
5.9 The ABB Structure	106

5.10 Summary of Structural Views.....	112
5.11 Instance Views.....	114
5.12 Preliminary ABB Operations	117
5.13 Preliminary Set of General Purpose ABBs (GPABBs)	122
Example 5.4 Matter Model (HIH Model, a.k.a. Linear Elastic Model)	123
5.14 Summary.....	124
CHAPTER 6 THE PBAM REPRESENTATION.....	125
6.1 Product Model Background	126
Example 6.1 Product Model (Surface Mount Resistor)	128
6.2 Product Model Transformations for Analysis.....	130
6.3 Simple PBAMs	133
Example 6.2 Simple PBAM (Component Extensional Model).....	135
6.4 Complex PBAMs	138
Example 6.3 Complex PBAM (Two Component Extensional Model)	139
6.5 PBAM Views.....	142
6.5.1 PBAM Structure	142
6.5.2 Constraint Schematic.....	144
6.5.3 Object Relationship Diagram	144
6.5.4 Subsystems.....	148
6.5.5 Extended Constraint Graphs	148
6.5.6 Input/Output Tables	148
6.5.7 Instance Views	148
6.6 Summary.....	148
CHAPTER 7 PRELIMINARY PBAM DEVELOPMENT GUIDELINES	150
7.1 Analysis Model Descriptions	150
7.2 PBAM Development Steps.....	151
7.3 Useful Development Skills	157
7.4 Discussion and Summary.....	158
CHAPTER 8 PRELIMINARY PBAM IMPLEMENTATION GUIDELINES	159
8.1 Philosophy	159
8.2 Implementing Structural Aspects Using Object-Oriented Programming	160
8.3 Implementing Relations Using Constraints.....	162
8.4 Summary.....	165

PART III VALIDATING THE PBAM REPRESENTATION

CHAPTER 9 PWA THERMOMECHANICAL ANALYSIS CASE STUDIES	167
9.1 Solder Joint Reliability Under Thermomechanical Loads.....	167
9.2 Case Study Solder Joint Fatigue Analysis Models.....	170
9.3 Development of PBAMs for Analysis of Solder Joint Fatigue	185
9.4 Implementation of Case Study PBAMs.....	197
9.5 Representative Design and Analysis Scenarios.....	198
9.6 Discussion of Solder Joint Fatigue Case Studies	209
9.7 PWA Warpage Case Study.....	212
9.8 Summary.....	216
CHAPTER 10 EVALUATION	217
10.1 Evaluation Approach.....	217
10.2 PBAMs Versus Objectives	220
10.3 General Discussion.....	240
10.4 Potential Sources of Discrepancies	242
10.5 Potential Benefits from Constraint Graph Theory	243
10.6 Implications for STEP.....	245
10.7 Summary.....	248

PART IV CLOSING REMARKS

CHAPTER 11 RECOMMENDED EXTENSIONS	250
CHAPTER 12 SUMMARY AND CONCLUSIONS	253

APPENDICES

APPENDIX A BACKGROUND MATERIAL	258
A.1 Overview of the Object Representation.....	258
A.2 EXPRESS-G Notation	262
A.3 IDEF0 Notation	263
APPENDIX B SOLDER JOINT FATIGUE ANALYSIS MODEL DESCRIPTIONS.....	264
APPENDIX C CONSTRAINT NOTATIONS	283
C.1 Extended Constraint Graph Notation	283
C.2 Constraint Schematic Notation	285
APPENDIX D PROTOTYPE CAD/CAE FRAMEWORK.....	291
D.1 General Architecture	291
D.2 PWA Product Modeling Tools	293
D.3 Finite Element Analysis Tool	295
D.4 Constraint Solver	295
D.5 Limitations of Prototype Implementation.....	295
APPENDIX E PWA PRODUCT MODEL.....	297
APPENDIX F ANALYTICAL BUILDING BLOCKS	299
F.1 Object Relationship Diagrams	299
F.2 General Purpose ABB Datasheets	311
APPENDIX G CASE STUDY PBAMS.....	320
REFERENCES	342
VITA.....	354

LIST OF TABLES

Table 2.1 Comparison of Representations	10
Table 2.2 Classes of Engineering Analysis	18
Table 2.3 Characteristics of Homogeneous and Heterogeneous Transformations.....	24
Table 4.1 Analysis Idealizations	43
Table 4.2 Inputs and Outputs of Analysis Idealizations.....	43
Table 4.3 Typical Analysis Model Options	50
Table 4.4 Natural Inputs and Outputs of Finite Element Analysis	53
Table 4.5 Some Categories of Analytical Relations.....	64
Table 5.1 Extended Constraint Graph Notation.....	73
Table 5.2 Basic Constraint Schematic Notation.....	79
Table 5.3 Mathematical Constraint Schematic Notation	81
Table 5.4 Constraint Schematic Inheritance Notation	87
Table 5.5 Basic Constraint Schematic Subsystem Notation.....	89
Table 5.6 Constraint Schematic Switch/Option Notation.....	96
Table 5.7 General Form of an I/O Table.....	104
Table 5.8 Example I/O Table for Elementary Rod	105
Table 5.9 ABB Structural Views.....	112
Table 5.10 Constraint Schematic Instance View Notation	114
Table 5.11 Categories of General Purpose ABBs (GPABBs)	122
Table 6.1 Representative STEP Parts for Product Models.....	127
Table 6.2 Characteristics of Simple and Complex PBAMs.....	138
Table 8.1 Mapping Analysis Models into the Object Representation	160
Table 9.1 Case Study Material Properties.....	177
Table 9.2 Strain Range Correction Factor	179
Table 9.3 Solder Joint Fatigue Case Study Analysis Results	204
Table 9.4 Example I/O Table for SJTF Model.....	207
Table 9.5 Solder Joint Parametric Study Using Implementation with Constraints.....	208
Table 10.1 Demonstration of Thesis Objectives in Case Studies	222
Table 10.2 Demonstration of Other Objectives in Case Studies	223
Table 10.3 Case Study Total Execution Times	233
Table 10.4 Breakdown of Execution Time	234
Table 10.5 Potential Libraries of Analysis Entities within STEP	247

LIST OF FIGURES

Figure 1.1 Cut-away View of a Printed Wiring Assembly	6
Figure 2.1 Analytical System Composed of Analytical Primitives.....	12
Figure 2.2 Typical Engineering Analysis Process.....	19
Figure 2.3 PWA Design Validation Process	20
Figure 2.4 Routine Analysis Using PBAMs	21
Figure 2.5 Resource Information for a General Design Task	23
Figure 2.6 Integration by Mapping between Neutral Schemas	24
Figure 4.1 Thesis Objectives for Analysis Model Representations.....	35
Figure 4.2 Other Objectives for Analysis Model Representations.....	36
Figure 4.3 Multiple Analysis Models for the Same Product	39
Figure 4.4 Steps in Routine Analysis	41
Figure 4.5 Challenges in Automated Analysis	42
Figure 4.6 Sample Printed Wiring Assembly (PWA).....	45
Figure 4.7 Product Model - Analysis Model Mapping	47
Figure 4.8 Linkages between Product Model and Analysis Model	47
Figure 4.9 Analysis Models of Varying Complexity Level and Regional Resolution	49
Figure 4.10 Typical Regions of Resolution in Electronic Packaging.....	57
Figure 4.11 Constraints Between a PWA and its Enclosure	58
Figure 4.12 Analysis Models from Diverse Disciplines for Assorted Applications	62
Figure 5.1 A Constraint Graph.....	68
Figure 5.2 Constraint Network for a D.C. Electric Motor	71
Figure 5.3 Equality Relation Notations.....	74
Figure 5.4 Example Constraint Graph	76
Figure 5.5 Example Extended Constraint Graph.....	76
Figure 5.6 Verbose Constraint Schematic of Object s	80
Figure 5.7 Constraint Schematic of Object s.....	80
Figure 5.8 A Simple Rod	82
Figure 5.9 Verbose Constraint Schematic for Elementary Rod	83
Figure 5.10 Constraint Schematic for Elementary Rod	84
Figure 5.11 Object Relationship Diagram for Elementary Rod.....	86
Figure 5.12 Constraint Schematic for Elementary Rod with Inheritance Notation	88
Figure 5.13 Some Views of an Object.....	89
Figure 5.14 Example Subsystem Views for Elementary Rod.....	91
Figure 5.15 An Exemplar Multibody Analysis Model	92
Figure 5.16 Two Views of an Analytical System	93
Figure 5.17 Subsystem Substitution Notation.....	99
Figure 5.18 Extended Constraint Graph for Elementary Rod.....	101
Figure 5.19 Special Case Extended Constraint Graph.....	101
Figure 5.20 Constraint Schematic for Safe Rod.....	102

Figure 5.21	Extended Constraint Graph for Safe Rod.....	103
Figure 5.22	Abbreviated Structure of an Analytical Building Block (ABB).....	108
Figure 5.23	General ABB Structure	110
Figure 5.24	ABB Structure of Elementary Deformable Body	111
Figure 5.25	ABB Structure of Elementary Rod	111
Figure 5.26	Representative Standard Instance View	116
Figure 5.27	Steps for Using ABBs in Routine Analysis	117
Figure 5.28	Example GPABB Object Relationship Diagram.....	123
Figure 5.29	A Matter Model	124
Figure 6.1	Classification of Analytical Building Blocks.....	125
Figure 6.2	Construction of a Thick Film Rectangular Chip Resistor.....	128
Figure 6.3	Simplified Product Model of a Surface Mount Resistor	129
Figure 6.4	Structure of a Simple PBAM.....	134
Figure 6.5	An Electrical Component Analysis Model.....	135
Figure 6.6	Constraint Schematic for Component Extensional Model	137
Figure 6.7	Example Subsystem View for Component Extensional Model	138
Figure 6.8	Structure of a Complex PBAM	139
Figure 6.9	Combined Elongation of Two Electrical Components.....	140
Figure 6.10	Constraint Schematic for Two Component Extensional Model	141
Figure 6.11	Example Subsystem View for Two Component Extensional Model	142
Figure 6.12	Abbreviated Structure of a PBAM.....	143
Figure 6.13	General PBAM Structure	145
Figure 6.14	PBAM Structure of Component Extensional Model	146
Figure 6.15	PBAM Structure of Two Component Extensional Model	147
Figure 7.1	Identification of Major Steps in Solder Joint Fatigue Analysis.....	152
Figure 8.1	Exemplar Implementation of a Constraint.....	163
Figure 9.1	Comparison of a) Through Hole and b) Surface Mount Technology	168
Figure 9.2	Major Steps in Solder Joint Fatigue Analysis	171
Figure 9.3	Cross Section of a Multilayer PWB/Component Assembly	173
Figure 9.4	Example Strain Model Variations.....	174
Figure 9.5	Example PWA Product Variations	175
Figure 9.6	Varying Levels of Thermomechanical Analysis Models.....	176
Figure 9.7	Level 1 Extensional Model.....	177
Figure 9.8	Level 3 Plane Strain Model	179
Figure 9.9	Idealized Solder Joint Geometry.....	180
Figure 9.10	PWA Analysis Model Object Relationship Diagram.....	186
Figure 9.11	Solder Joint Thermomechanical Fatigue Model Instance View.....	189
Figure 9.12	Solder Joint Thermomechanical Fatigue Model Subsystem View	190
Figure 9.13	Extended Constraint Graph of SJTF Model under Thermal Cycling	192
Figure 9.14	Extensional Model Instance View.....	193
Figure 9.15	Extensional Model Subsystem View	194

Figure 9.16	Plane Strain Model Constraint Schematic	195
Figure 9.17	PBAM Commonality and Modularity	196
Figure 9.18	Prototype Implementation of Case Study PBAMs	199
Figure 9.19	Standard Instance View of SJTF Model	202
Figure 9.20	Solder Joint Reliability Checker Using PWA Two Rod Model.....	203
Figure 9.21	Deformations with Rectangular Solder Joint Geometry Option (Case #62)	205
Figure 9.22	Shear Stress Distribution in Detailed Solder Joint (Case #3)	206
Figure 9.23	Example Subsystem View for SJTF Model.....	207
Figure 9.24	Conceptual PBAM for Analysis of PWA Warpage	213
Figure 9.25	Study of Global / Local Analysis Model Interaction.....	214
Figure 10.1	Simplified Version of Modeling Process	218
Figure 10.2	Routine Analysis Steps Automated Using PBAMs.....	228
Figure A.1	Example of Object and Class Concepts.....	258
Figure A.2	Example Class Hierarchy.....	261
Figure A.3	Basic EXPRESS-G Notation with Extensions	262
Figure A.4	Extended IDEF0 Notation.....	263
Figure D.1	Object-Oriented Engineering Information System	292
Figure D.2	View of PWA Information in the Objectbase	294
Figure E.1	Ad-hoc PWA Product Model.....	298
Figure F.1	Product Model-Analysis Model Associations	301
Figure F.2	Analytical Primitives	302
Figure F.3	Matter	303
Figure F.4	Matter Models	304
Figure F.5	Slender Bodies.....	305
Figure F.6	Discrete Primitives.....	306
Figure F.7	Analytical Variables	308
Figure F.8	Analytical Systems.....	309
Figure F.9	PBAMs for PWA Analysis Applications.....	310

SUMMARY

In spite of recent advancements in computer aided design and engineering (CAD/E), such as parametric geometry and automatic mesh generation, a large gap exists between computer-based design models and analysis models. Transforming a detailed product design into an engineering analysis model can require large amounts of heterogeneous information, engineering theory, and industrial heuristics. Consequently, traditional computational methods alone are insufficient for the effective automation of this process.

This research defines a new representation of analysis models, termed *product model-based analytical models* (PBAMs), which enables the automation of a particular class of analysis problems. Specifically, PBAMs automate the instantiation, execution, and interaction of a variety of *routine analysis models* (i.e., models that are used repeatedly during product design). This research also defines *constraint schematics* - one of the first known combinations of constraint graph theory and object-oriented concepts for the purpose of representing engineering analysis models. Constraint schematics are one of several views of the PBAM representation. Preliminary guidelines for both developing and implementing PBAMs of routine analysis models are included that utilize these views.

PBAMs enable rapid, flexible, routine analysis by linking detailed design models with potentially many analysis models of varying complexity and application. Furthermore, this representation aides exploration of "what if" design problems by supporting reversible input/output combinations in some cases. The PBAM representation has been evaluated using case studies in the thermomechanical fatigue of solder joints on printed wiring assemblies (Engelmaier, 1983, 1989; Lau, et al., 1986). Results show that PBAMs uniformly represent such analysis models containing a mixture of both formula-based and finite element-based relations.

PART I PROBLEM DEFINITION

CHAPTER 1

INTRODUCTION

*I have more than enough to do than to extract my own FEA data.
I am talking to manufacturing, tooling, vendors, fatigue and fractures,
the stress group - I simply don't have time.*
Design Engineer, Airplane Structures
[Liker, et al., 1992]

Engineering analysis is an important step in the product design process which serves to evaluate design alternatives against various product criteria. Depending on the design phase and the kind of answers sought, an analysis may involve a qualitative evaluation based on rules of thumb or a detailed numerical calculation based on fundamental physical laws. The typical goal of such engineering analyses is to predict the behavior of a system without prototyping and observing the actual system.

Engineering Analysis Benefits

Because of this goal, some have used the term "virtual prototyping" to describe general computer modeling, including computerized analysis [Puttre, 1992b]. Puttre reports that the advantages companies expect from "virtual prototyping" include:

1. *Reduced product development time and cost.*

He reports how one company cut their product development time in half and eliminated physical prototypes after incorporating finite element analysis into their design process.

2. Improved product designs.

Computer-based engineering analysis can help designers explore many different design alternatives that might otherwise be cost and time prohibitive. Furthermore, once a design alternative has been selected, analyses can guide design refinement to meet product requirements.

3. Better analysis results.

Puttre [1992b] notes that certain physical phenomena are not readily understood by testing physical prototypes. Similarly, some product operating conditions are impossible to simulate physically in laboratory environments. Thus, engineers are using computer-based analysis to study phenomena such as nonlinear fluid mechanics and to simulate conditions like those encountered by submarines.

Engineering Analysis Issues

In spite of the above potential benefits, computer-based engineering analysis during product design is not as widespread as one might imagine. Four observations from the industrial experience of the author regarding such analysis are as follows:

1. Determining what type of analysis model to use for a given design situation is often complicated by inexperience and unawareness of proven analysis models that exist in the literature or corporate experience-base.
2. Creating an analysis model for the design problem at hand can be very time consuming. Usually there is no computer-based associativity between the detailed design model and the needed analysis model.
3. Collecting the information necessary to define the analysis model (material properties, simplified geometry, load conditions, etc.) and formatting it for input into available solution tools can be quite laborious.

4. Judging the quality of the subsequent results is hindered by inexperience, by improper understanding of the limitations of the analysis model used, and by lack of computer-based associativity between the analysis model and the product model.

Organizations that develop consumer or business products, such as cars, computers, and telephones, often cannot afford to support many highly trained, full time analysts; therefore, design engineers are often left to perform engineering analyses along with a multitude of other tasks. Because of the above barriers, designers may inadvertently obtain invalid results, may expend too many resources in performing an analysis, or may even fail to perform an analysis due to constraints such as schedule deadlines.

These observations were confirmed in a recent industry survey [Linker et al., 1992] which concluded that, when it exists at all, the computer aided engineering (CAE) department is separate from design and often understaffed. For example, they report that one large auto company has over 600 computer aided design (CAD) users but only 12 CAE analysts. This quote from a consumer products design engineer conveys the realities of resource limitations:

The senior analyst is so busy, he has seven to eight months lead time. I can build a prototype and test it faster than CAE and feel confident it is accurate. Mostly junior people are assigned to CAE and lack job experience.

Design Engineer, Consumer Products Co. [Liker, et al., 1992]

Thrust of Thesis

This thesis addresses these kinds of problems through the use of engineering information management techniques and artificial intelligence concepts. In a general sense this research is aimed at improving the design of complex products by providing new methods of cooperation between engineer and computer in the solution of analysis problems.

However, enabling generalized seamless integration between design and analysis is considered to be too lofty a goal at present. Therefore, this research has focused on automating what is termed *routine analysis* - the design and verification of products using established analysis models. (This term and others are defined more fully in Chapter 2) Routine analysis models, then, are reasonably well understood and are meant to be applied over and over during the product design process.

Hence, the thrust of this research has been the development of a new analysis model representation, called *product model-based analytical models* (PBAMs), which provides rapid, flexible routine analysis capabilities for use concurrent with product design. To provide rapidity, PBAMs are linked with detailed design data. To provide flexibility, this new approach can represent multiple analysis models of varying complexity and application for the same type of product.

Representative Application Area

The analysis of thermomechanical behavior in printed wiring assemblies (PWAs) has been chosen as a representative application area within which to illustrate and validate the new concepts. Figure 1.1 illustrates how a PWA is basically a complex composite assembly from a mechanical engineering viewpoint. Thermomechanical problems resulting from the mismatch in coefficients of thermal expansion (CTEs) between printed wiring board (PWB) layers and electrical components recently have become more pronounced due to the increased power densities and decreased package sizes typical of modern PWAs [Fulton and Ume, et al., 1990].

The design of a PWA involves the processing of large amounts of heterogeneous information by multidisciplinary teams which include electrical, manufacturing, mechanical, and systems engineers. The analysis of PWA thermomechanical behavior

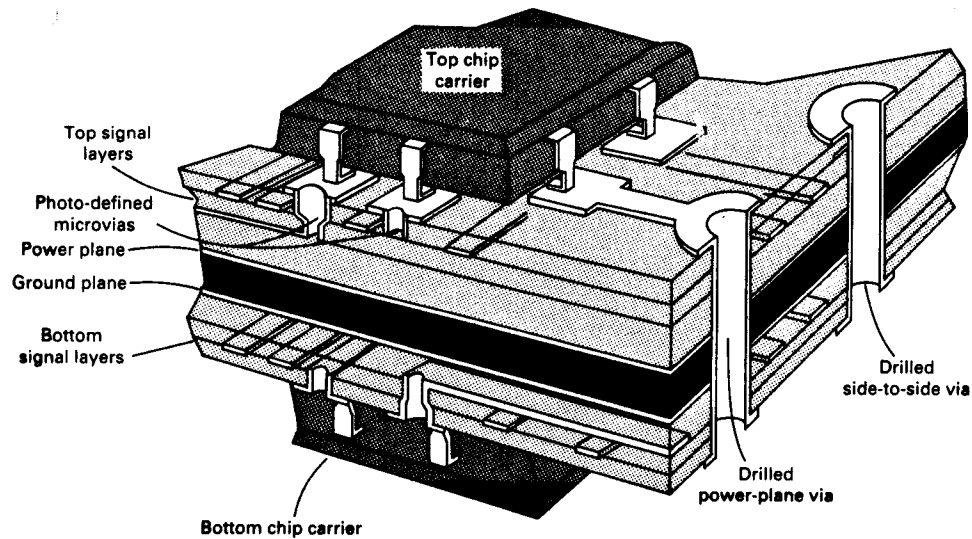


Figure 1.1 Cut-away View of a Printed Wiring Assembly
[Pinnel and Knausenberger, 1989]

typically involves the evaluation of subsets of this heterogeneous information against various design criteria. Therefore, the use of PBAMs in the analysis of such products serves to validate and demonstrate the relevance and capabilities of the PBAM representation.

Organization of Thesis

This thesis is organized into four major parts¹. Part I defines the problem addressed by introducing the topic (Chapter 1) and defining different classes of engineering analysis in Chapter 2, both from a process viewpoint and from an information integration perspective. PBAMs also are introduced at a high-level in that chapter to show how they can help automate routine analysis. The present state of engineering analysis practice and research is then reported in Chapter 3 to highlight gaps in current

¹To give a historical perspective, note that Chapters 1, 2, 3 and Appendices A, D, E are largely based on an earlier paper [Peak and Fulton, 1992b]. Chapters 4, 10 and Appendix F include extensions of that work as well.

capabilities. Chapter 4 identifies capabilities that ideally would be part of any automated analysis strategy and specifies which new capabilities have been the priorities in this thesis.

Part II contains the major contribution of this research. It begins in Chapter 5 by introducing the analytical building block (ABB) representation, constraint schematic notation, and underlying constraint graph and object concepts. Chapter 6 shows that the PBAM representation is a special case of the ABB representation that includes linkages between product information and analysis information. Chapter 7 provides preliminary PBAM Development Guidelines, i.e., how to map a typically unstructured description of a specific analysis model into a PBAM representation of that model. Finally, Chapter 8 contains preliminary PBAM Implementation Guidelines that discuss how to map from the PBAM representation of a particular analysis model into an object-oriented, constraint-based implementation.

Specific case studies that test and illustrate the capabilities of the PBAM representation are in Chapter 9 of Part III. Validation continues in Chapter 10 by evaluating how well PBAMs meet the thesis objectives first given in Chapter 4. General discussion is also included in this chapter. Finally, the thesis concludes with Part IV by recommending extensions to the research (Chapter 11) and summarizing the contributions of thesis (Chapter 12).

CHAPTER 2

PROBLEM DESCRIPTION

This section describes the specific problem addressed in this research within the context of general engineering analysis automation. Introductory definitions are given first, followed by a review of the engineering analysis process itself. Then approaches to automating analysis will be given to show the need for representations of analysis models. PBAMs are then introduced to the extent of how they can be used to automate analysis during design.

2.1 Introductory Definitions

As engineers work with many different kinds of models, it is important to understand which models are dealt with in this thesis. Therefore, a few preliminary definitions are given first to give a basis for the proceeding problem description. It should be noted that these definitions represent one of the first attempts (albeit possibly incomplete and in need of further refinement) to define the intersection between the familiar, mature, and well-defined analysis model world, and the somewhat foreign, newer, and evolving product model world (defined below). It has been rather challenging to conceptualize the merger of these two diverse worlds. Thus, these definitions at a minimum are an attempt to establish a starting point from which further discussion and research can evolve.

DEFINITION 2.1 An **analytical model** is an engineering approximation of physical behavior in exact form.

For example, consider a coil spring in an automobile suspension, and assume one wishes to estimate its compression when the car is at rest. One simple analytical model of this physical behavior (deformation of the spring) is the following equation:

$$u = W / k \quad (2.1)$$

Here W is the proportion of the automobile weight supported by the spring, k is the spring constant, and u is the deflection of the spring (how much it compresses). Thus, the mathematical equation is the exact form of this analytical model. Note, however, that the true physical behavior of the spring is much more complicated than can be expressed by just these three parameters. The above equation is only an approximation of the physical behavior.

DEFINITION 2.2 An **analysis model** is an analytical model or an approximation of an analytical model.

Per this definition, the above spring model is also an analysis model. If one needs more detailed information about the behavior of the spring, another possible analytical model is a 3D elasticity model. This analytical model would have a complicated boundary value problem (BVP) as its exact mathematical form. Approximation techniques are typically used to find the solution to such an analysis model when no exact solution is known to exist. For example, the finite element method would yield a finite element model (FEM) that is an approximation of the analytical model (i.e., the FEM is not exact in comparison to the precise mathematical BVP). Thus, a FEM is an example of an analysis model that is not an analytical model.

As a starting point to address the needs identified in Chapter 1, the following terms deal with the description of analysis models (both analytical and non-analytical) in a form that can be computerized.

DEFINITION 2.3 A **representation** is a computable approximation of "reality" for an intended purpose.

For comparison purposes, Table 2.1 shows that the "reality" addressed by geometric representations is the shape of physical objects. There can be more than one way to represent this "reality" including wireframe and boundary representations. The intended purpose of a representation determines what kind of information must be included in the representation.

Table 2.1 Comparison of Representations

	Analysis Model Representation	Geometry Representation
	Analysis Model	Physical Object
"Reality" Example Representations	Finite Element Models	Wireframe
	Bond Graphs	Constructive Solid Geometry (CSG)
	PBAMs	Boundary Representation (B-rep)

A representation has a defined structure and defined operations. For example, the structure of the constructive solid geometry (CSG) representation is a binary tree with solid primitive shapes and operators as its nodes [Mortenson, 1985 p. 372]. CSG operators are union, difference, and intersection.

An important point regarding a representation is that it is computable, i.e., it can be implemented in a computer. To emphasize this point, the following term from the Standard for the Exchange of Product Model Data (STEP) [ISO 10303-1, 1992] is helpful: an **information model** is "a formal (being in accordance with rules explicitly established prior to use) model (an abstract description) of a bounded set of facts, concepts or instructions to meet a specified requirement." The term representation has been used above instead of the term information model due to the related use of the

former in computational geometry, as well as to emphasize both the structure and operations of the information. However, both terms can be used interchangeably in this thesis.

DEFINITION 2.4 The **analytical building block (ABB) representation** is a representation of analysis models. (This thesis defines the structure and operations of this representation.)

A specific ABB represents a specific analysis model (e.g., an ideal spring). Thus, analysis models are the "reality" in the ABB representation analogous to physical object shape in geometric representations (Table 2.1). In the name of this representation, "analytical" conveys that the representation has an exact form, while "building block" means that a specific ABB can be used to build other ABBs and so on.

DEFINITION 2.5 A **product model** is a representation of a product (e.g., a physical system, assembly, or part) that contains product life cycle information.

Life cycle information is that information needed by all parties associated with a product, including design, analysis, manufacturing, installation, repair, marketing, and management. Such information includes geometry, bill of materials, assembly instructions, and test specifications. Currently no such computer-based, omniscient product model exists, but steps are being taken in the NIST PDES / ISO STEP project described later towards such a product model [ISO 10303-X]. *For the purposes of this thesis, the term **product model** will be used in a narrower sense to mean the representation of design-oriented product information in order to distinguish it from analysis-oriented representations.*

DEFINITION 2.6 An **analytical primitive** is an ABB representing analysis concepts such as basic continua, discrete elements, state variables, and material models.

In the above order, examples include ideal beams, ideal springs, displacements, and linear elastic stress-strain models. Product information is not included in an analytical primitive.

DEFINITION 2.7 An **analytical system** is an ABB that is a collection of analytical primitives.

As with analytical primitives, product information is not included in an analytical system. Figure 2.1 illustrates that analytical primitives (distributed load, beam, and linear elastic material) can be assembled to form an analytical system (an Euler beam system).

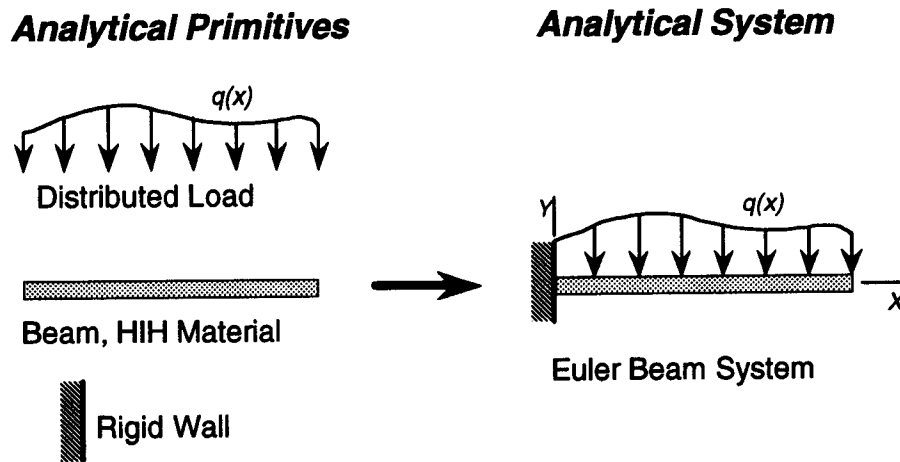


Figure 2.1 Analytical System Composed of Analytical Primitives

DEFINITION 2.8 A **product model-based analytical model (PBAM)** is an ABB that is a collection of other ABBs and linkages with a product model.

Thus, the **PBAM representation** (defined in this thesis) is a special case of the ABB representation. In simpler terms, *a PBAM is a representation of an analysis model*

that includes linkages with product design information. An ideal spring alone could be represented as an analytical primitive. However, the representation of an ideal spring and how it relates to the product model of the real physical spring can be achieved using a PBAM (as will be shown). Note that a PBAM can be built from other ABBs, including analytical primitives, analytical systems, and other PBAMs.

The above terms will be used in the remainder of the thesis to communicate precisely what model is being referred to at any given time.

2.2 The Engineering Analysis Process

The engineering analysis process can be defined as solving engineering problems by "the analytical prediction and interpretation of physical phenomena, for the purpose of providing a basis for all manner of engineering decisions involving design, evaluation, and use of equipment" [Ryder, 1961, p. 1]. In what they term the *professional method*, Ver Planck and Teare [1954, p. 28] identify five stages of the engineering analysis process:

Stage 1. Define the problem

Collect and analyze the facts in relation to the original question in order to fully discover and define the problem.

Stage 2. Plan its treatment

Determine what values, principles, attitudes, and basic practices are applicable to the problem. Plan the means of dealing with the facts in the light of these ways of approach.

Stage 3. Execute the plan

Carry through the plan so as to reach a decision or result. (Often the decision does not end the problem but clarifies or changes the issue so that the problem is started over in a new aspect.)

Stage 4. Check the work as a whole before using the solution

Go over the results, first systematically, then realistically in terms of use, and at last with reference to the general knowledge and experience of that field.

Stage 5. Learn and generalize if possible

Take thought to find what can be learned that is of value in the situation at hand and what can be learned that may be of use in future problems.

They also identify four functions that may be performed throughout the process as a whole as well as at each of the above stages:

Function 1. Simplification

- a. By restricting assumptions.
- b. By condensed, exact statement in words or in symbols.

Function 2. Alternation between analysis and insight.

Function 3. Checking validity, both systematically and realistically in terms of use.

Function 4. Using all that can be learned by experience as the solution proceeds as a basis for correction and as a guide to future steps, even if this involves radical change in the problem or its treatment.

They note that engineering analysis is a *deductive* process (applying general principles to the specific design problem at hand), where as the scientific method which is *inductive* (seeking to find general principles from observation of specific cases). Numerous case studies showing how to apply their professional method to "real engineering problems" are included. As opposed to textbook problems that are geared towards aiding the learning of a particular topic, their problems are "real" in the sense that they are representative of those encountered by practicing engineers. Such problems usually are not neatly setup and typically do not have a single clean answer.

In a book with a similar bent towards teaching the process of solving "real" engineering problems, Ryder summarizes his universal technique for problem solving in the following six steps [Ryder, 1961, p. 7]:

1. The problem is recognized and defined.
2. A distinction is made between the essential and the nonessential phenomena, and the latter are ignored, or else set aside for later consideration.
3. Idealizations and simplifications are made which will render the problem manageable without making the results misleading.
4. The significant parameters are noted and the physical relationships among them are expressed in mathematical form.
5. The relationships are manipulated to yield significant mathematical results.
6. The mathematical results are interpreted in terms of the physical conditions which they represent.

The authors of both books note that the steps they define are not always sequential but involve strategies of high-level overviews that gain more detail with iterative decomposition (much like the design process itself). Backtracking and modification are also noted as frequent occurrences during the development of a new analysis model.

Planck and Teare identify two extremes of analysis in engineering practice [1954, p. 1]:

1. Day-to-day application of previously developed analysis models.
2. Development of new analysis models for new products and/or new situations.

Though not specifically labeled as such by them, these extremes are analogous to the variant and adaptive/original classes of design, respectively, defined by Pahl and Beitz [1988]. Brown and Chandrasekaran [1984] categorize design problems in a similar manner with the terms Class 1, Class 2, and Class 3 design. Class 1 (original) design involves developing a new solution principle to perform a new or known task (e.g., creating an automobile power source beyond fossil fuel engines and electric engines).

Class 2 (adaptive) design adapts a known system to accomplish a changed task (e.g., designing a bicycle powered lawnmower). Finally, Class 3 (variant or routine) design varies the size and/or arrangement of some aspects of a chosen system to perform the same basic function (e.g., developing screwdrivers with the same basic shape, but with varying sizes). Such distinctions have proven helpful in focusing and categorizing design automation research. For example, rather than trying to tackle the whole spectrum of design, Brown and Chandrasekaran [1984] concentrate on strategies to aide Class 3 design.

In a similar vein, the following types of analysis models are defined now to clarify the problem addressed in this thesis.

DEFINITION 2.9 A **routine analysis model** is an established analysis model that is repeatedly used on a specific type of product.

For example, Engelmaier has developed an analysis model which estimates the fatigue life of solder joints on a PWA [1983, 1989]. He defines which types of components the analysis model is valid for (e.g., surface mount chip resistors), and he specifies which idealized geometric parameters and material properties to use. Apparently his analysis model is used commonly in industry today, as evidenced by its inclusion in handbooks (e.g., [p. 158, Hinch, 1988]), conference tutorials (e.g., [SMT CON, 1991]), and technical courses (e.g., [SMT DFM, 1992]). Product designers simply can use this analysis model by plugging in data specific to their own PWAs; they do not have to develop their own analysis model to predict solder joint fatigue life. (As this analysis model is used as a case study described later, the complete articles [1983, 1989] are included in Appendix B).

Routine analysis is the *process* of employing routine analysis models, which emphasizes repeatedly *using* proven analysis models on new product instances and on design iterations for the same product instance. The variables and relations used in a routine analysis model are known a priori and are relatively constant. In contrast, the *value* of each variable (e.g., the length of a resistor) is not known beforehand but is dependent on the specific product instance being analyzed.

The term routine is not meant to imply that the analysis model(s) involved are simplistic. After an analysis model has been developed and its utilization in design is understood, even the most sophisticated analysis model could be considered a routine model. Understanding the limitations of the analysis models and knowing how to apply the results to design are the primary skills required of a user of routine analysis models.

DEFINITION 2.10 An **adaptive analysis model** is a new analysis model developed by adapting some aspect of an existing routine analysis model.

For example, if an existing routine model supports only steady state loads, one could investigate the effects of time-varying loads to see if any particular advantages are gained (e.g., better correlation between analysis results and experimental results). Thus, the emphasis of this class of analysis is to *extend* an analysis model, not just use it. Validating the resulting new analysis model using representative product data is also part of the adaptive analysis process. A specific example of adaptive analysis would be extending Engelmaier's analysis model to include the effects of conformal coating (a resin often applied over a whole PWA to improve reliability under severe environmental conditions [Pound, 1989]).

Table 2.2 Classes of Engineering Analysis

Class	Task Performer	Task	Task Output
Routine	Product Designer	<i>Use</i> established analysis model repeatedly.	Analysis Results, Design Changes
Adaptive	Product Designer or Engineering Analyst	<i>Extend</i> routine model for same product type.	Extended Analysis Model, Sample Results
Original	Engineering Analyst and Experimentalist	<i>Develop</i> new analysis model for same / new product type.	New Analysis Model, Sample Results

DEFINITION 2.11 An **original analysis model** is an entirely new analysis model that:

1. Replaces or supplements existing analysis models for a given product type, or
2. Analyzes a new problem for a new or existing product type.

This type of analysis almost always would require correlation with physical experiments to ensure its validity. This class of analysis focuses on analysis model development rather than repeated analysis model use and typically requires the most engineering intellect. The process of creating Engelmaier's original analysis model could be classified as an example of original analysis. Creating analysis models to aide in the design of micromachines [Kiriya, 1993] is another example as neither the product nor the analysis techniques are well defined.

Table 2.2 summarizes the above discussion and includes the person who typically might perform each class of analysis. Note that the output of adaptive and original analysis can be an analysis model which becomes a routine analysis model after widespread use. These categories are not dogmatic and some overlap may exist.

2.3 Automating Routine Analysis - Process Viewpoint

Using the above definitions of the engineering analysis process as a basis, Figure 2.2 shows a typical engineering analysis process using IDEF₀ notation (see Appendix A) which shows its context in the product design process [adapted from MSC, 1990]. Note that it includes iteration on the design itself, as well as iteration on the analysis model as mentioned above.

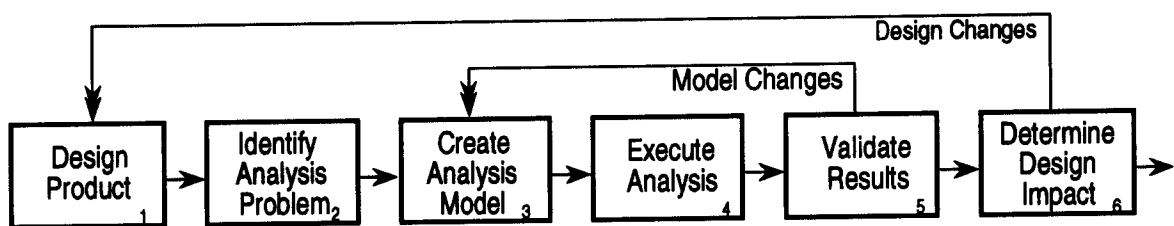


Figure 2.2 Typical Engineering Analysis Process

Figure 2.3 gives a specific example of this generic process during the board layout stage of the PWA design process. At this stage detailed component types are selected and their location on the board is determined. Also, the board outline, parent assembly interfaces, and cooling needs, are typically determined at this stage. The figure is based on previous work by Yeh and the author of this thesis in who developed a detailed IDEF₀ process model of the PWA design process [Fulton, Ume, et al., 1990].

Each product requirement to be checked may have its own analysis model or series of analysis models to judge if the preliminary PWA design is acceptable. It is important to note that numerous analysis models exist that can be used to check such PWA requirements. For example, Mentor Graphic's Autotherm computes component temperatures to check component reliability. Lau [1991] contains numerous solder joint fatigue models (though not all are oriented towards design use). Steinberg [1988] devotes a whole book to electronic equipment vibration analysis. Garratt [1993] describes work

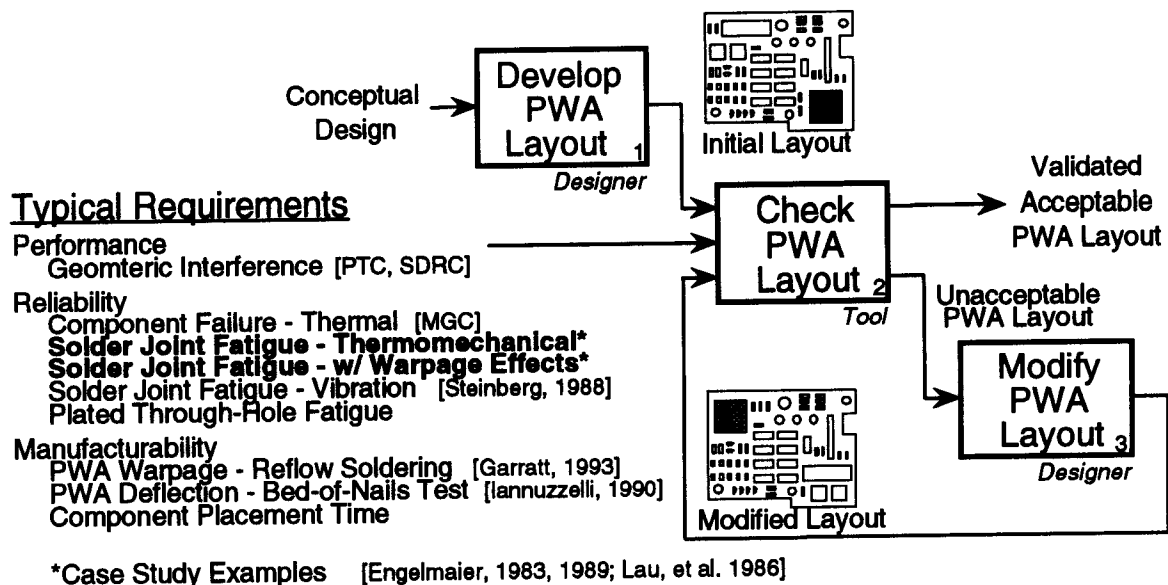


Figure 2.3 PWA Design Validation Process

initiated in the Advanced Electronic Packaging Lab (AEPL) at Georgia Tech to simulate thermomechanical deformations in a simplified bare printed wiring board (PWB) during the reflow soldering process. Furthermore, Iannuzzelli [1990] describes a suite of analysis models to simulate the behavior of a PWA during various manufacturing processes, including bed-of-nails testing.

The results obtained from such analysis models may indicate the need for a design change. For example, if the design is deemed unacceptable from a component reliability point of view, the component could be moved to a cooler area on the PWA (as shown in modified PWA layout in Figure 2.3), or the enclosure could be modified to provide more cooling to that area of the PWA. Resources permitting, this modified PWA design should then be re-checked.

These examples show how a variety of checks needs to be made during the design of a product. For a given product type (e.g., PWAs), the same types of analysis models typically need to be created and executed for each new product instance and at several

stages during the design process. In such cases it would be helpful to have pre-defined catalogs of such routine analysis models that can be used after being populated with the specific data of a new design as needed.

Without going into the details of the PBAM representation yet, the potential benefits of such a routine analysis model representation will now be discussed. The point of view taken is how one would use a catalog of routine analysis models after they have been represented as PBAMs and implemented in a computing environment.

With checking solder joint fatigue as an example context, Figure 2.4 illustrates a

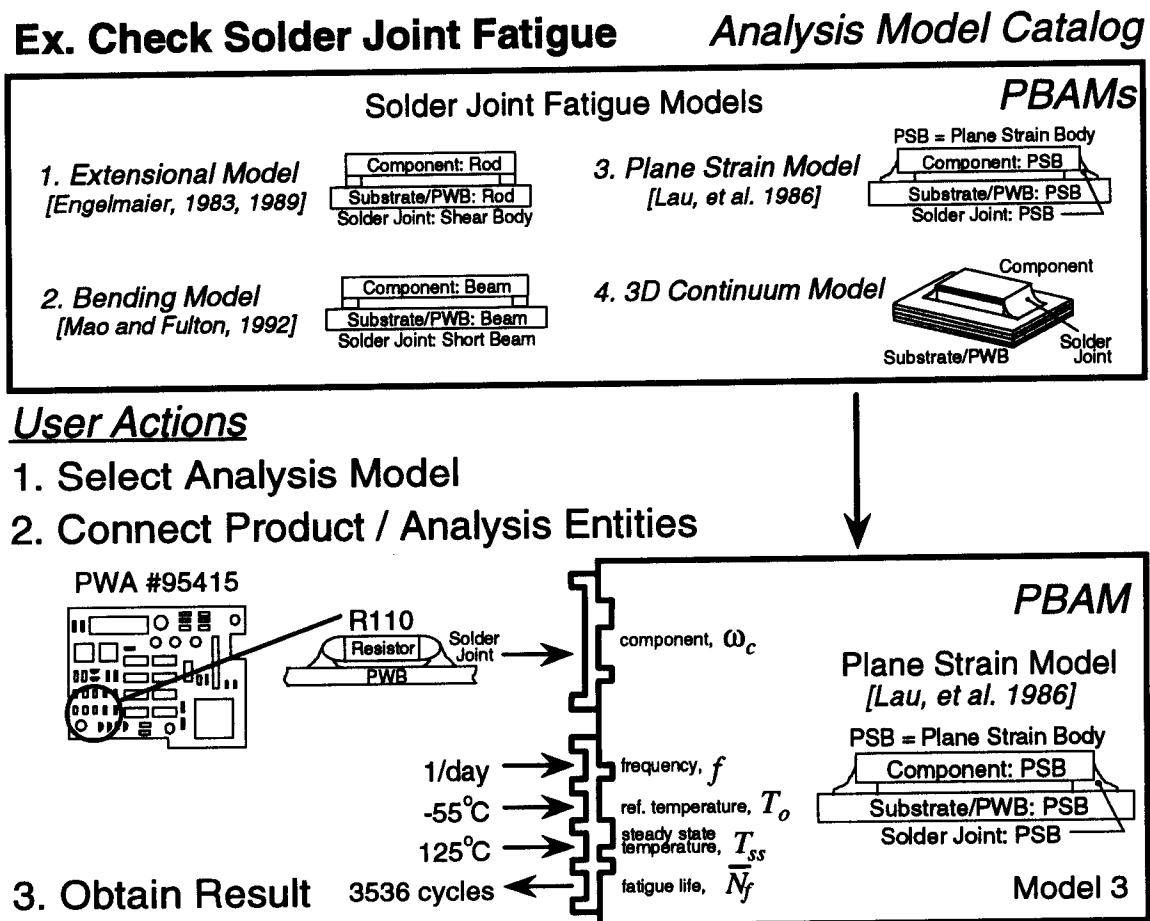


Figure 2.4 Routine Analysis Using PBAMs

PBAM usage scenario. Depending on the design stage and problem being addressed, analysis models of varying complexity and computational cost could be needed and would be part of the catalog as illustrated. First, the user selects which analysis model (represented by a PBAM) to use. Second, the required product and analysis entities are connected to the selected PBAM at a high level. In Figure 2-2 the component of interest (R110), the load frequency, and the temperature extremes are such inputs. The PBAM automatically extracts detailed information from the connected product and analysis entities to create the analysis model it represents. The creation, execution, and interaction of submodels within the analysis model (if applicable) are also handled automatically. Finally, the PBAM allows the user to obtain the results in a form that is meaningful to the problem at hand. A solder joint fatigue life of 3536 cycles is the result shown in the figure.

Per this example, one can view a PBAM as a black box to which the user inputs product and analysis information. Analysis results are then obtained without the user having to deal with all the detail inside the analysis model representation. In a nutshell, from a process viewpoint, this thesis addresses the problem of representing analysis models in order to automate routine analysis.

2.4 Automated Routine Analysis - Integration Viewpoint

As another perspective of automating routine analysis, this research also addresses the information integration problem of accessing certain types of *resource information*, i.e., building blocks for engineering design. Some classifications of resource information are shown in Figure 2.5. Dym and Levitt [1991] offer an alternate "typology of engineering knowledge" which is broader and more abstract than the categorization given below.

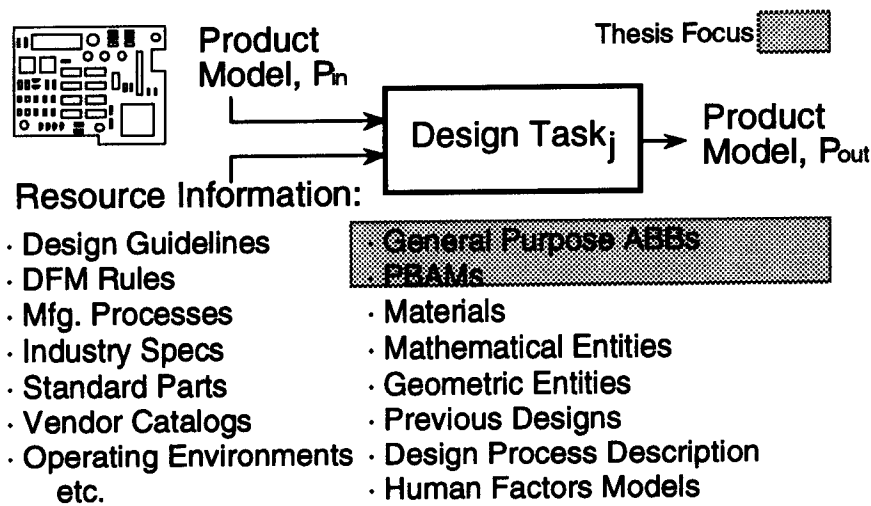


Figure 2.5 Resource Information for a General Design Task

To automate any design task (including analysis), resource information needed to perform the task must be accessible. Therefore, a prerequisite to design task automation in general is the representation of such resource information. Hence, the creation of a general representation of routine analysis models addresses a subset of this information integration problem.

A paraphrase of one popular definition of engineering information integration is *providing the right information at the right time in the right form to perform the right task*. Often this information provision requires a distinct process step that transforms information from one form into another. In general this transformation can be viewed as a mapping between information schemas (i.e., information models or representations) [Rangan, 1992; Rosen 1992].

Per the ANSI 3-schema architecture [see, e.g., Bray, 1988, p. 60; Yang, 1991] Figure 2.6 shows how common neutral schemas can be used to exchange information between various CAD/CAE tools that may have their own proprietary internal schema.

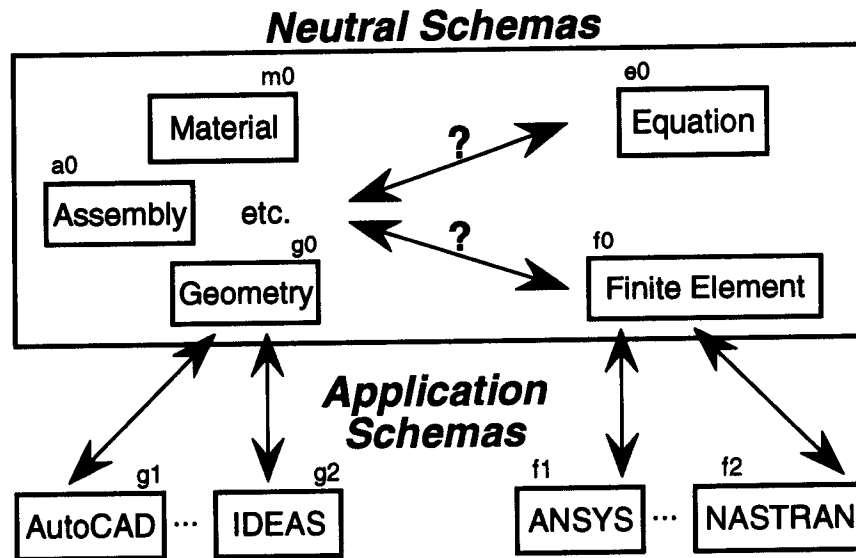


Figure 2.6 Integration by Mapping between Neutral Schemas

However, the similarity between the source and target schemas greatly influences the complexity of the mapping process. For example, exchanging a geometric model between two geometric modeling tools (e.g., AutoCAD and SDRC IDEAS), or exchanging two finite element models between finite element analysis tools (e.g., ANSYS and NASTRAN), though not trivial, is a simpler problem than transforming a product model into an analysis model. Characteristics of these two different types of mappings, termed **homogeneous** and **heterogeneous** for convenience, are given in Table 2.3.

Table 2.3 Characteristics of Homogeneous and Heterogeneous Transformations

Mapping Type	Example	Amount Constrained		Dependency on Attribute Values
		Forward	Inverse	
Homogeneous	$g1 \leftrightarrow g0 \leftrightarrow g2$	\approx Fully	\approx Fully	Low
Heterogeneous	$(g0, m0, \text{etc.}) \leftrightarrow f0$	Under	Highly Under	High

Defining intermediate representations that fit somewhere in between product representations (e.g., of detailed geometry) and general purpose analysis representations (e.g., of finite element models) is one approach to helping bridge the gap between heterogeneous schemas. The PBAM representation can be viewed as one such intermediate representation.

2.5 Summary

This chapter defined different types of models that are dealt with in this thesis, including analysis models, product models, and PBAMs (which are representations of analysis models). The engineering process in general and the routine analysis process in particular was described. Routine analysis models were identified as the class of analysis models addressed by the PBAM representation. Then a preview was shown from a user view of how PBAMs could help automate the routine analysis process. Finally, the design-analysis information integration problem was reviewed. A brief summary of the problem addressed in this thesis follows:

A representation of routine analysis models is needed that bridges the integration gap between product information and analysis information.

This thesis is an attempt to define such a representation, which is called the PBAM representation.

CHAPTER 3

RELATED WORK

Other research efforts in the area of automated analysis are now described, and their relations to this research are noted. Organization is roughly based on the analysis process model given in Figure 2.2.

3.1 Automated Modeling in Engineering Analysis

A general theme noted by Rosenberg and Redfield [1988] is that the generation of solutions once an analysis model has been defined is a fairly well-developed field. Thus, one major area of research in automated analysis is **automated modeling**, i.e., the generation of models by computer assisted/automated means.

3.1.1 General Industrial Practice

The first step in a typical modeling process employed in industry today is *manually* creating an abstracted/simplified model of the detailed product scenario. Geometric and behavioral simplifications are made (e.g., neglecting small radii, assuming linear stress-strain relations, and assuming a pure planar state of stress). It is *after* this step that a tool like MSC/XL [MSC, 1990] is used to capture this *simplified* model for subsequent discretized finite element analysis. Alternately, if the analysis model is simple enough, the mathematical boundary value problem model may be developed and solved manually or automatically in a tool like *Mathematica* [Wolfram, 1991]. Thus a large gap exists between the detailed product data model and the analysis model using current practices. Mentor Graphics' PWA thermal analysis tool, Autotherm [MGC, 1991b], offers an

example of limited association between detailed product model and analysis model. However, such specialized tools typically suffer from limited extendibility and applicability.

3.1.2 Automated Modeling in FEA

Finite element analysis (FEA) is perhaps the most commonly used analysis technique in industry today. Much work currently is aimed at automating element selection and mesh generation [Mackerle and Osborn, 1988]. Typically the prototype systems reviewed query the user to obtain necessary input information (e.g., PLASHTRAN [Cagan 1987]). SACON, in a review by Mackerle and Osborn [1988], starts one step back and helps the user select a FEA strategy.

Shephard and his colleagues at Rensselaer Polytechnic Institute have done extensive work in the area of design and analysis integration [e.g., Shephard, et al, 1988, 1990, 1992; Niu and Shephard, 1990; Wentorf, et al., 1992]. In general their work appears to be aimed more towards the automation of adaptive and original analysis, with a focus on the geometric aspects of finite element-based analysis models.

For example, Shephard and Finnigan [1988] describe an approach for integrating geometric models and finite element models. Their non-product model view requires that attributes (e.g., material properties) be added to the geometric model in preparation for finite element (FE) modeling. They conclude that the generalized simplification of product geometry for FE modeling purposes and the use of multiple element types remain open issues.

With an extensive background in automatic mesh generation, Shephard and associates evidently have approached the design and analysis integration problem starting from FEA and moving towards design. As this thesis has started at the opposite point,

there exists a good potential for extensions of this thesis to take advantage of their work, and possibly vice-versa.

3.1.3 General Research Efforts

Dym and Levitt [1991] give a broad view of the use of engineering knowledge in design and analysis. They claim that mathematical and numerical representations alone, though essential, are inadequate, and that other means of representing engineering knowledge are needed. They note the value of search as a problem solving technique, as proposed in Chapter 2, but offer no application of it towards engineering analysis.

The Finite Element Analysis Critic in CASE [Rehg, et al., 1988] is intended to perform the modeling, solution, and feedback tasks of Figure 2.2, but it is not clear how it actually works.

Mashburn and Anderson [1991] propose the idea of constructing analysis models for kinematics from primitives; however, the linkage between model primitives and detailed product data is not mentioned.

An "intelligent modeling environment" implementation (ATHENA Aide) is described by Fink, et al. [1988] in which the detailed situation (in this case a power plant) is first modeled by an analyst using discrete idealized power plant components (WestinghousePump, WestinghousePressurizer, etc.). These components, which are represented using objects, are models that are in between detailed product data and generic analytical primitives, thus, triggering this research towards the notion of PBAMs which act as semantic handles for the designer. In these authors' opinion "the initial abstraction of a [analysis] model for the physical system of interest is a task that must remain with the analyst."

A series of papers by Powell and An-Nashif [Powell and An-Nashif, 1988; An-Nashif and Powell, 1989, 1991] describe a similar approach for automated the modeling of frame structures which starts with an intermediate application-specific model based on their component-connection representation. As with Fink, et al., the analysis model thus generated is used as input to a FEA system but is not linked to a detailed product model.

3.1.4 Problem Solving Using Libraries of Primitives

The definition of a finite set of primitive components for use in building complex systems is a decomposition approach to problem solving that is quite common in engineering. For example, the Initial Graphics Exchange Specification (IGES) defines geometric primitives that can be combined into complex geometric models [NIST, 1990]. The development of libraries of product modeling primitives that range from generic to specific applicability is the basic concept behind the layered architecture found in the **Standard for the Exchange of Product Model Data (STEP)** [Yang, 1991]. Analogously, this thesis is an attempt to define a representation that could be used to create libraries of analysis models.

This idea of libraries of analysis models has been indirectly proposed by Dodds, et al. [1982] in their suggestion of an "engineering mechanics machine." Parisi and Rehab [1986] describe an implementation of a "probability virtual machine" for use in engineering computations. The *Mathematica* system [Wolfram, 1991] and the differential equation system of Peskin and Russo [1988] are examples of systems with libraries of mathematical primitives.

3.1.5 Libraries of Analysis Entities within STEP

Much of the CAD/CAE effort of the past 20 years has focused on the geometric description of products. Recently attention has been given to including non-geometric information such as material, assembly information, cost, and versioning; thus, the

emergence of product models. STEP (the **ST**andard for the **E**xchange of **P**roduct **M**odel **D**ata) is an effort headed by the International Standards Organization (ISO) aimed at providing standard representations of product data (i.e., standard libraries of product-related entities) [ISO 10303-X; Ryan, 1992]. The initial focus of this effort is the exchange of product data between heterogeneous CAD/CAE tools. PDES (**P**roduct **D**ata **E**xchange using STEP) is a U. S. *activity* aimed at accelerating the development of the STEP *standard*.

The entities defined in STEP contain attributes which can be inherited by subclasses but have only loose behavioral association. Therefore, the current STEP approach appears to be based more on the AI notion of frames (also known as schemas [Charniak and McDermott, 1985]) rather than full-fledged objects.

Representation of some types of resource information, such as materials and finite element models, also is being addressed in STEP. The Reference section in this thesis includes a complete listing of the Parts (i.e., a collections of related entities) within STEP as of the writing of this thesis. The numbers indicate the assigned Part number (e.g., Part 104 defines entities related to finite element analysis). Thus, STEP is actually a collection of standards where each Part is a standard that may utilize other Parts. A complete listing of STEP Parts as of this date is included in the References at the end of this thesis.

Only Part 11 (The EXPRESS Language) has completed the standardization process to date, while several other Parts are Draft International Standards (DIS). Many Parts are in a lesser draft status and have a long ways to go, so it remains to be seen how effective these standards will be in actual industrial practice.

3.1.6 Design Synthesis Using Analysis Models

Redfield and Mooring [1988] describe the use of analysis models (bond graphs) as a starting point for synthesizing specific products which have the desired behavior of the analysis model. This approach is useful for conceptual design as it lets one explore many design alternatives to achieve the desired behavior. Luth, et al. [1991] and Jain, et al. [1991] propose a similar approach for mapping from function to form and use idealized application-specific entities (Hanger, FloorPlate, etc.) in the conceptual structural design of floor framings. No linkage to the subsequent detailed product design is indicated.

3.2 Analysis Model Generation Using Product Model Data

Many acknowledge the importance of providing access to the product design information required for an analysis [Dym and Levitt, 1991; Korngold, et al., 1991], but apparently no one has yet shown how generally to represent analysis models with linkages to the detailed product data.

3.3 Representation of Analysis Problems

Dixon, et al. [1988] define design problems in terms of initial design states and desired states. They consider analysis as a kind of "assessment" problem, which can be classified according to the type of analysis required and the level of certainty desired.

Similarly, an analysis is identified as a special case of "verification" by Chandrasekaran [1990] where verification is one kind of design subtask. He suggests the idea of attempting to solve a problem first with qualitative analysis. The subsequent qualitative results may indicate the need for a detailed quantitative analysis. Shepherd [1988] focuses on analysis problems in particular and

... addresses the issues of specifying the functional information beyond that of the geometric domain of the object needed to qualify a problem in mathematical physics for which an engineering analysis is to be performed.

He defines a geometry-based approach to specify this information.

Mistree, et al., [1990] have developed Decision Support Problems (DSPs) as part of their Decision-Based Design paradigm. DSPs appear to be definitions of analysis problems with a strong design process and optimization perspective. Bras [1992] in particular has looked at "designing design processes," where in one sense defining the analysis problem is defining the design process which initiates, performs, and uses an analysis.

3.4 Representation of Analysis Models

3.4.1 Requirements of Analysis Model Representations

Breedveld [1988] defines necessary characteristics for an analysis model using the bond graph representation. As bond graphs have been primarily used to date in modeling discrete physical system dynamics, emphasis is placed on connectivity characteristics including rules for connecting primitives. However, links with product data are not mentioned. Other objectives for analysis model representations are identified in the next chapter.

3.4.2 Quantitative Representations

Dym and Levitt [1991] discuss the appropriateness of various knowledge representations from the field of artificial intelligence for general engineering applications. Below is a list of several current analysis model representation schemes. Included is a qualitative judgment of their shortcomings with respect to application in automated routine analysis.

1. Mathematical Equations: Lack of semantic handle.
2. Procedural Languages: Poor reusability.
3. Bond Graphs: Lack of semantic handles.
4. Relational Tables: Awkward association with behavior.
5. Finite Elements: Little capture of analysis intent.

Additionally, all the above representations suffer from a lack of association with the detailed product/situation that is being modeled. With the possible exception of bond graphs [Rosenberg and Karnopp, 1983; Ingram, 1989a], they do not easily support inheritance of attributes and behaviors. Capture of derivational information that is needed to express assumptions and limitations of the analysis model is also not achieved.

3.4.3 Qualitative Representations

DeKleer and Brown, Forbus, and Kuipers describe the fundamental concepts of "qualitative reasoning about physical systems" in a book by the same title [Bobrow, ed., 1984]. Fruchter, et. al [1991] describe the application of such concepts to the design and analysis of civil engineering structures. Kiriya [1992, 1993] has defined analytical building block-like entities with qualitative behavior using Forbus's qualitative process theory for a variety of engineering applications.

Qualitative analysis models have not been pursued in this thesis. They are viewed in a broader sense as another type of analysis model which fit in at the lowest point on the analysis model accuracy scale. However, their utility cannot be overlooked since design rules that require non-numeric evaluations and explanation of results can potentially take advantage of qualitative analysis models.

3.5 Summary of Gaps

Considering the preceding literature review, the following two items stand out as having received little attention:

- General approaches for linking detailed product models and engineering analysis models.
- Concurrent representation of multiple analysis models of varying complexity and applicability for the same product.

Therefore, the claim in Chapter 2 that there is a need for a representation that addresses these gaps is apparently justified. The next chapter discusses these issues in more detail and identifies the capabilities that an analysis model representation should have.

CHAPTER 4

OBJECTIVES FOR ANALYSIS MODEL REPRESENTATIONS

*Better one handful with tranquillity
than two handfuls with toil and chasing after the wind.*
Ecclesiastes 4:6 [NIV]

The previous two chapters defined the general problem of automating routine analysis and identified some of the gaps in the current state of automated analysis in general. This chapter defines the capabilities that an ideal analysis model representation should have to address the automated routine analysis problem and related gaps. The primary purpose here is to define the targets at which the PBAM representation has been aimed. These targets can be used as criteria for evaluating potentially any proposed analysis model representation, as is done in Chapter 10 for the PBAM representation.

The new capabilities that are the focus of this thesis are discussed in the first section of this chapter and are called THESIS OBJECTIVES (Figure 4.1). Other capabilities

1. Represent Routine Analysis Models
2. Automate Routine Analysis
3. Provide Product-Analysis Associativity
4. Represent Multiple Complexity Levels
5. Support Analysis Model Options
6. Be Modular / Seamless
7. Enable Rapid Analysis
8. Be Flexible
9. Enable Multiple Input/Output Directions

Figure 4.1 THESIS OBJECTIVES for Analysis Model Representations

that any ideal analysis model representation should fulfill are labeled OTHER OBJECTIVES (Figure 4.2) and are described in the second section. These categories are defined to emphasize which new capabilities have been given priority in this thesis. Though an attempt has been made to define orthogonal, measurable objectives, some objectives overlap and some are related. In several cases, an overall objective (e.g., Objectives 2, 8, and 9) is decomposed into other objectives as identified within the description of each objective.

Support Desired Capabilities:	Support Dimensions of Analysis Model Diversity:
10. Provide Final Results 11. Provide Intermediate Results 12. Enable Interaction of Analysis Models 13. Allow Multivalued Inputs 14. Provide Multivalued Outputs 15. Allow Solution Procedure Control 16. Allow Global/Local Models (Resolution) 17. Support Analysis Model Exchange 18. Utilize Existing CAE/CAD Tools 19. Encapsulate Existing Specialized Design-Analysis Linkages	20. Purposes 21. Product Domains 22. Product Types 23. Applications 24. Disciplines 25. Behavior Regimes 26. Solution Methods 27. Variables 28. Relations 29. Systems of Relations

Figure 4.2 OTHER OBJECTIVES for Analysis Model Representations

A number of these objectives were identified as needed capabilities through discussions with engineers having extensive analysis experience (e.g., D. Agonafer of IBM-Poughkeepsie, J. I. Craig of Georgia Tech-Aerospace Engineering, R. E. Fulton of Georgia Tech-Mechanical Engineering, and C. P. Yeh of Motorola-Schaumburg). Some needed capabilities were identified by the author in the course of carrying out the case

studies in this thesis (e.g., OBJECTIVES 1, 4, 5, 12, 15, and 25). The need for other capabilities are self evident if one considers the types of analysis models commonly encountered in undergraduate engineering (e.g., Objectives 10-11, 18-19, and 26-29). Where possible, references from the literature are given that further substantiate the need for a particular capability. In some ways this chapter itself could be considered a minor contribution of this thesis as it is one of the first known attempts to identify a comprehensive list of desirable analysis model representation capabilities.

4.1 THESIS OBJECTIVES

OBJECTIVE 1 Representation Provide a representation of analysis models which meets all other THESIS OBJECTIVES and many of the OTHER OBJECTIVES.

Many of the OTHER OBJECTIVES can be met today using disjoint and often ad-hoc approaches which are difficult to implement and extend. Because of this situation, this objective includes bringing together as many of those capabilities as possible into one uniform representation.

The following list containing characteristics of a good knowledge representation is based on criteria set forth by Rich [1983] and Winston [1984]. A good representation of knowledge should:

- a. *Have a well-defined structure composed of a pre-defined vocabulary of symbols.*

The logical pieces of the representation should be defined as well as rules on how those pieces can be assembled together.

- b. *Have a defined method for being developed from the specific real world entities being represented.* For the representation to be useful, guidelines for mapping analysis model descriptions into the representation (Development Guidelines) must be provided.

- c. *Be capable of expressing any relevant aspects of the content of the real world input.*
In other words, it should be complete with respect to its intended purpose. With respect to the analysis model world, the "relevant aspects" are those needed to meet the THESIS OBJECTIVES (1-9) and OTHER OBJECTIVES (10-29) identified in this chapter. Being able to package these capabilities into one representation is itself viewed as a major objective of this research, though not all capabilities are fully supported.
- d. *Be easily modified.* This criteria is covered by OBJECTIVE 6 (Modularity / Seamlessness).
- e. *Be understood by the people who are familiar with the knowledge being represented.*
Engineers should be able to comprehend the representation of an analysis model with which they are familiar. Ideally, the same engineers should be able to develop a representation of a particular analysis model by using the Development Guidelines just mentioned in item (b.).
- f. *Be explicit and transparent.* The important features of an analysis model should be easily recognizable in the representation. For example, problem geometry and material properties should be identifiable, as well as the relations among such parameters. One representation that does not meet this criteria would be a neural network representation of analysis models. It would be considered an implicit representation where the important features would be distributed among all the synaptic nets; thus, they would be difficult to access and identify.
- g. *Hide detail unless expressly requested (i.e., support encapsulation).* To perform a specific task using the representation, one should not have to wade through a large amount of irrelevant information.
- h. *Be concise.* This criteria is somewhat subjective, but can be taken to mean "contains only the necessary information."

- i. *Be computable.* Guidelines for transforming an analysis model representation into specific computerized implementation forms (Implementation Guidelines) must be given. For example, important algorithms should be expressed in pseudo code or flow charts that have been tested in a specific computer language.
 - j. *Be efficient.* Several measures of efficiency can be defined including run time, computer memory requirements. OBJECTIVE 7 (Speed) deals with run time efficiency.
- Another way to view this objective is as an attempt to capture analysis models in a computer at a higher level of intent than currently exists. For example, a typical finite element model has little, if any, explicit information about how its attributes (e.g., material properties, loads, idealized geometry) are derived from the product it is used on. Figure 4.3 illustrates how multiple analysis models can be associated with the same product type, including how solution-method-specific discretized analysis models like FEM could be derived from a PBAM (Finite Element Model 1.2.1) as illustrated. Finite Element Model 1.2.2 might have a finer finite element mesh than 1.2.1 in order to judge solution

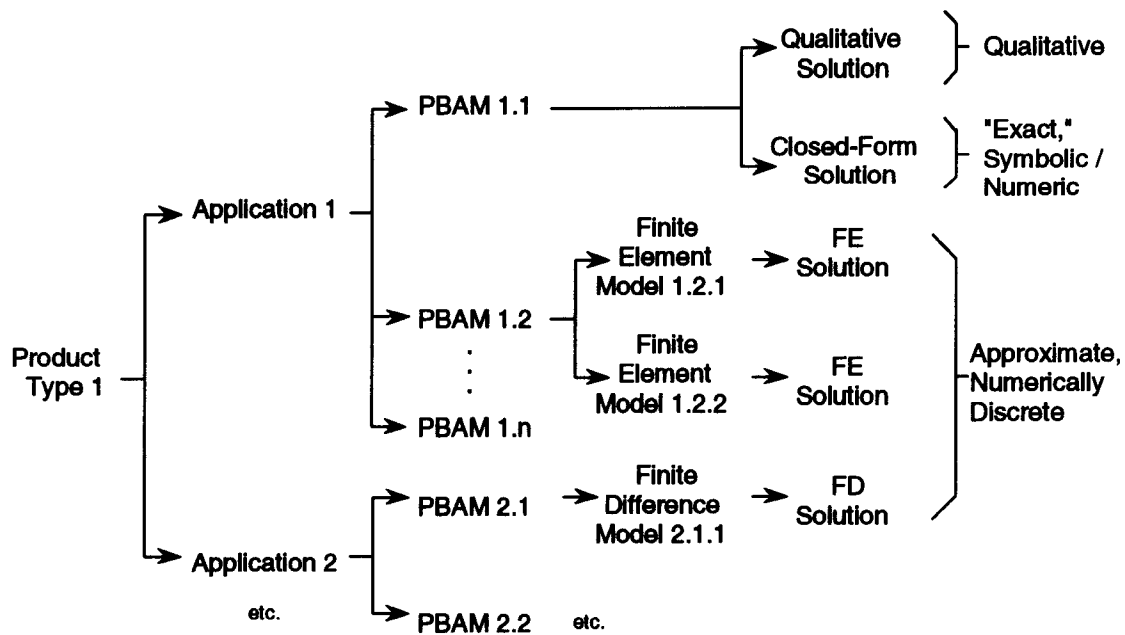


Figure 4.3 Multiple Analysis Models for the Same Product

convergence. Thus, the higher-level intent of a PBAM can also enable the derivation of multiple lower-level models.

OBJECTIVE 2 Automation Fully automate routine analysis.

To make this objective measurable, the routine analysis process has been modeled as having the steps given in Figure 4.4 (based on the discussion in Chapter 2). Steps 1 and 3 are design process steps that provide input to and receive output from routine analysis and are shown for context. In this thesis, priority has been given to the case where routine analysis is used for the purpose of design verification (Objective 20 Purpose).

As such, focus has been given to Steps 2.3 through 2.6. Hence, this objective with respect to the PBAM representation is to alleviate the human engineer from tedious, error-prone tasks. The engineer is still needed to understand what is going on inside an analysis model representation, at least to the extent that the limitations of the analysis model are understood. In other words the engineer is expected to know what design problems need addressing (Step 2.1) and which analysis models are appropriate for different circumstances (Step 2.2). Similarly, he or she must check the analysis results (Step 2.7) and apply the results appropriately (Step 2.8).

1. Design product.
2. Perform routine analysis.
 - 2.1. Identify application (design problem).
 - 2.2. Choose analysis model(s) for problem.
 - * 2.3. Link analysis model(s) with product & analysis inputs.
 - * 2.4. Execute (solve) analysis models(s).
 - * 2.5. Manage interaction of analysis models.
 - * 2.6. Extract results.
 - 2.7. Check results.
 - 2.8. Interpret results to determine design changes.
3. Make design changes.
 - * Research Priority

Figure 4.4 Steps in Routine Analysis

Rosenberg and Redfield [1988] have noted that tools to automate some aspects of product design (Steps 1 and 3) and many aspects of individual analysis model solution (Step 2.5) are fairly well developed and could be thought of as "islands of automation." From this perspective, this objective is to build bridges between these islands.

Note that a routine analysis model, by definition, does not require *creation*, as that would require adaptive or original analysis. Rather, it requires *instantiation*, i.e., populating a pre-defined template with product and analysis information that is specific to the problem at hand (Step 2.3). Therefore, automating routine analysis is viewed as a more realistic objective than automating engineering analysis in general.

Still, there are many challenges associated with automated routine analysis as illustrated in Figure 4.5. As indicated, meeting this objective involves fulfilling Objectives 3, 4, 9, 12, and 26 in particular which are described in this chapter.

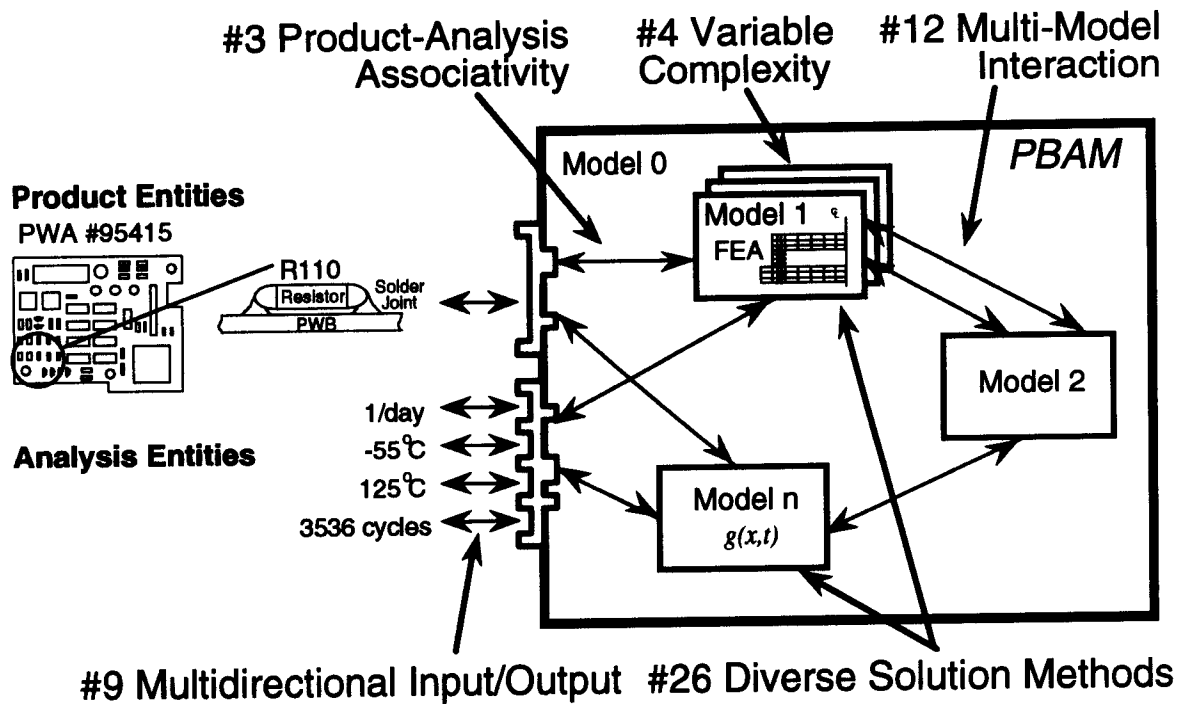


Figure 4.5 Challenges in Automated Analysis

OBJECTIVE 3 Associativity Link analysis models with product models.

To achieve the automation objective just described, the product data must be linked with the analysis data to provide input to the analysis model and to receive outputs from it. These linkages are called **product-analysis transformations (PATs)** for future reference (Chapter 6). Without such linkages, someone must manually perform these time-consuming and error-prone information transfers.

The representation should support information flows in two directions (corresponding to the forward and inverse forms of a PAT): 1) from product model to analysis model and 2) from analysis model to product model. Characteristics of the **analysis idealizations** and **design synthesis operations** that occur in these directions, respectively, are now given.

Analysis Idealizations

An **analysis idealization** is a transformation that simplifies the physical (product) situation into analysis attributes [after Shephard, et al. 1990, 1992]. Idealizations have been categorized by Finn, et al. [1992] as summarized in Table 4.1.

Table 4.1 Analysis Idealizations

<u>Category</u>	<u>Example</u>
• Dimensional Reduction	• Assuming deformation occurs in only one dimension.
• Geometric Symmetries	• Assuming loads and behaviors are symmetric.
• Feature Removal	• Neglecting fillets and small holes.
• Domain Alteration	• Assuming an aerofoil to be a thin plate.
• Phenomena Removal	• Considering only thermal behavior.
• Phenomena Reduction	• Assuming thermal behavior does not include radiation

Because an analysis idealization is not typically as under constrained as a design synthesis operation, and because it supply inputs to analysis models, an analysis idealization is considered to be the *natural direction* of a PAT, while a design operation is called the inverse direction of a PAT. Based on the categories in the previous table, Table 4.2 lists typical inputs and outputs of analysis idealizations, which are natural inputs and natural outputs of PATs per the preceding terminology.

Table 4.2 Inputs and Outputs of Analysis Idealizations

<u>Inputs</u>	<u>Outputs</u>
• performance specifications	• simplified system parameters
• systems, assemblies, parts	• body or region parameters
• detailed geometry	- simplified geometry
• materials	- material properties
	- loads
	- boundary conditions

Design Synthesis Operations

Design synthesis operations generate product design information by adding detail to analysis outputs or transforming them. As they can be thought of as the inverse of analysis idealizations, the inputs and outputs in Table 4.2 reverse roles in design operations. These operations are typically under constrained as illustrated by the following examples.

Ex. Select a PWB material with $CTE = 6 \times 10^{-6} \text{ (in/in)/}^{\circ}\text{C}$ to match the CTE of a ceramic component [see Engelmaier, 1983].

Ex. Determine a beam cross section where $I = 0.5 \text{ in}^4$.

Ex. For improved reliability, determine the (x, y) location of a component on a PWA so that its temperature will be less than 100°C during operation [MGC, 1991b].

Ex. Select a gull wing component for increased flexibility versus a leadless component to give improved fatigue life. Peak and Fulton [1992a] discuss electronic component selection in general.

In each case, the design parameter (material, cross section, (x,y) , and component) has many possible choices to meet the given condition. The details of how each such design operation is carried out is beyond the scope of this thesis. Therefore, the focus of this objective with respect to this thesis is to have the analysis model representation determine the inputs to such design operations (i.e., determine analysis idealization outputs, a. k. a. analysis model natural inputs - see Objective 9 Directionality). Then, at a minimum, the designer can find the desired design operation output by iteratively supplying candidate design parameters and running the analysis to see if they meet the given condition.

Example

The issues and usefulness of such linkages are best illustrated by an example. Consider the PWA shown in Figure 4.6. The product model of this part (i.e., its detailed definition: geometry, electrical connectivity, etc.) can be created to some degree by an electrical CAD (ECAD) tool such as BoardStation by Mentor Graphics Corporation.

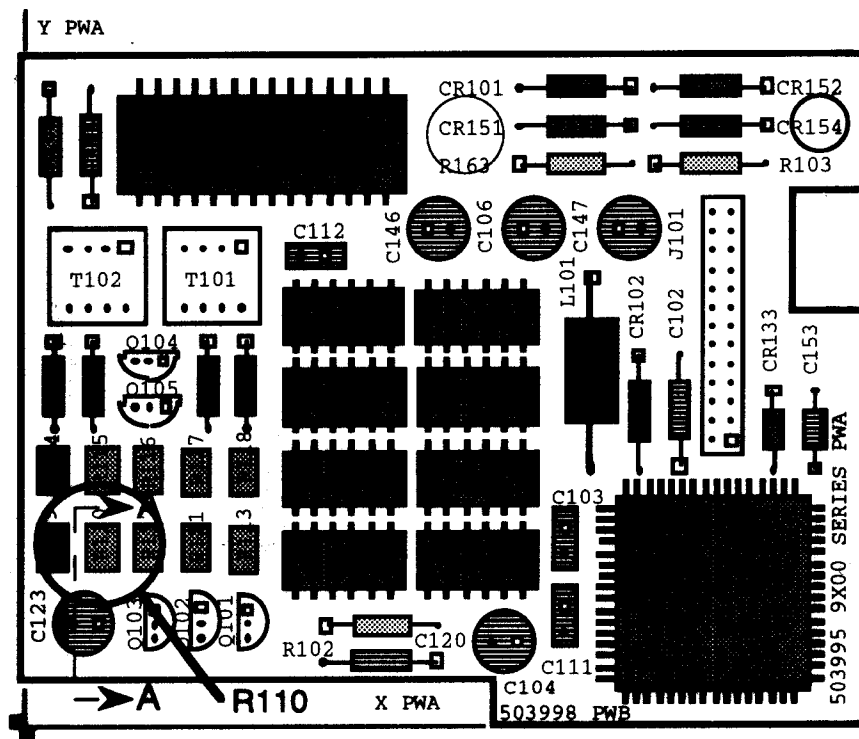


Figure 4.6 Sample Printed Wiring Assembly (PWA)

If one desires to determine the fatigue life of a component's solder joints, for example surface mount resistor R110, then as a first approximation Engelmaier's model [Engelmaier, 1983, 1989] (described in Appendix B) could be used. A cross-section of the component - solder joint - PWB assembly is given in Figure 4.7 along with a corresponding analysis model that is represented as an analytical system constructed from

analytical primitives. Only extensional deformation is considered for the resistor and PWB so they are modeled as rods. Simple shear is the only assumed mode of deformation in the solder joints to compute their fully relaxed shear strain, $\Delta\gamma$, for use later in the Coffin-Manson fatigue equation.

Figure 4.8 shows the information linkages between the analysis model and the product model via product-analysis transformations (PATs), Φ_i , with the R110-Rod1 linkage given in detail. Note that the resistor, R110, and the PWB are themselves complex assemblies and that the product model contains much detailed information that the analysis model does not use (e.g., toleranced dimensions). Consequently some transformations are performed on the product data to simplify it for use in the analysis model (e.g., a simple nominalization function performs the transformation Φ_{31}).

ECAD tools typically do not capture all the information that is needed to perform even this relatively simple analysis. For example, functional product data, such as the expected number of power-up cycles over the life of the product or the ambient operating temperature, needs to be included in the product model. Similarly, the details of a typical solder joint shape might be needed for more complex analyses and could be included in a representation of the soldering process.

After the solution to this analytical system has been obtained, it is then used in another relation and finally the resulting fatigue life estimate, N_f , must be fed back into the product model.

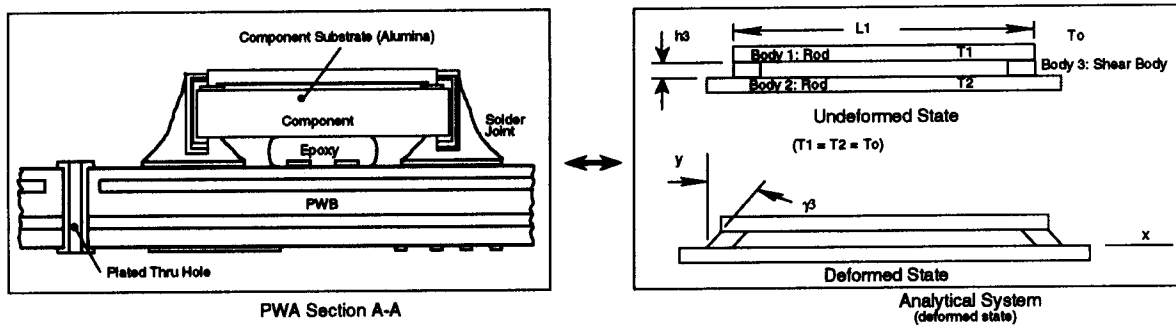


Figure 4.7 Product Model - Analysis Model Mapping

Product Model Entities

Deformation Problem 1

ref. temperature: 25 °C
 steady state temp.: 125 °C
 strain range: (To Be Determined)

PWB 1

Resistor 110

part number: 99120
 value: 47 Ω +/- 5%
 wattage: 1/8 W
 price: \$0.007
 components: [
 (Substrate
 material: Alumina
 geometry: ...)
 (Barrier Layer
 material: Nickel
 geometry: ...)
 ...]
 total length: 0.220 +/- 0.010"
 total width: 0.070 +/- 0.006"
 total thickness: 0.020 +/- 0.004"

Solder Joint 1

Analytical System Entities

Interconnected Rods System 1

ref. temperature:

Φ_{11}

Φ_2

Body 2: Rod

Φ_3

Body 1: Rod

Φ_{31}

length: (total length) nominal.

Φ_{32}

area: (total width * total thickness) nominal.

Φ_{33}

stress-strain model: (substrate material) as H1H material.

Φ_{12}

temperature: (steady state temp.)

Φ_4

Body 3: Shear Body

Φ_{13}

shear strain: (TBD)

Figure 4.8 Linkages between Product Model and Analysis Model

OBJECTIVE 4 Complexity Level Represent analysis models of varying complexity (e.g., topology, geometry, material model) for the same regional resolution.

Figure 4.3 also illustrates how multiple analysis models of varying complexity levels can be associated with the same analysis application. Depending on the design need, a simple analysis model with a formula-based solution (PBAM 1.1) might suffice, or a more detailed analysis model (PBAM 1.2) might be used which requires a more costly finite element solution. Consequently, it is felt that supporting analysis models of varying complexity levels should be supported by an analysis model representation. Kiriya [1993] and Catley, et al., [1991, p. 198] concur with this objective of having analysis models of varying complexity for different stages in the design process (with respect to micromachine design and ship design, respectively). The description of the case studies in Chapter 9 contains examples of solder joint fatigue analysis model variations.

Analysis models having the same **regional resolution** (i.e., covering the same basic geometric area), might vary in terms of **compositional topology** (i.e., number of distinct bodies). Within a given body, complexity variation also can occur in terms of geometric detail, behavior dimensions (e.g., extension (1D) vs. plane strain (2D) deformation), and constitutive relation (e.g., linear elastic vs. bilinear plastic material models). This concept is illustrated for determination of solder joint strain in Figure 4.9 where analysis models of increasing complexity at the same regional resolution (Region 2) are labeled Level 1 through Level 4.

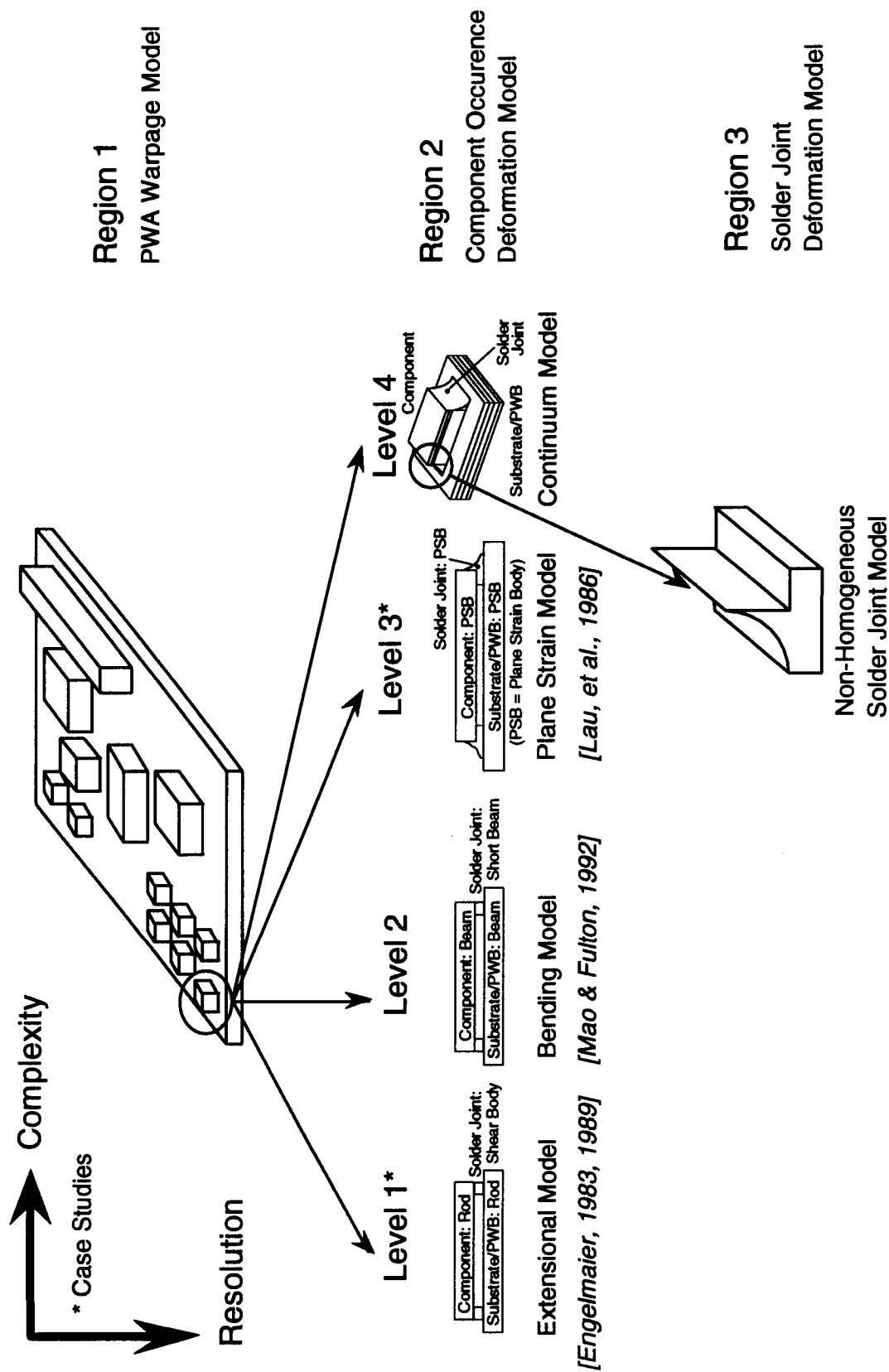


Figure 4.9 Analysis Models of Varying Complexity Level and Regional Resolution

OBJECTIVE 5 Options Allow choices in analysis model operation.

For increased flexibility, an analysis model representation should include options that the user can select to specify variations in analysis model behavior. Categories of such options include complexity level (OBJECTIVE 4), different load conditions, and whether or not to include different effects (Table 4.3).

Table 4.3 Typical Analysis Model Options

<u>Option Type</u>	<u>Example</u>
• loading option	• steady state vs. transient loads
• complexity variations	• geometric detail, stress-strain model type
• additional effects	• warpage effects
• subsystem substitutions	• extensional vs. plane strain analysis models

As a further example, suppose one is interested in the fatigue life of a structure under a cyclic load. A typical analysis model for this case could be divided into two major submodels that perform the following steps: 1) determine the maximum stress in the structure (using a structural analysis model), and 2) determine the fatigue life based on that stress and material properties (using a material fatigue model). One possible option category for this case would be the structural analysis model used (e.g., a simple rod, or a complex body supporting 3D stress-strain states).

Such options may not be dependent on internal analysis model factors, but rather may be determined by external factors like the accuracy needed at a particular design stage. As discussed in Objective 4 (Complexity Level) a simple formula-based structural analysis model might be appropriate in the early stages of design. Factors effecting the selection of this option could include the desire for quick answer (with low computational cost) which does not have to be very accurate. Using such external factors to select analysis model options is not considered in this research. However, supporting analysis

model options at least would give the user something to choose from and, hence, would make an analysis model representation versatile and adaptable to different design needs.

OBJECTIVE 6 Modularity/Seamlessness Provide modularity to represent new analysis models with minimal impact on analysis models already represented.

Modularity of knowledge representations has been defined to mean "the ability to add, modify, or delete individual data structures more or less independently of the remainder of the database, that is, with clearly circumscribed effects of what the system 'knows' " [Barr and Feigenbaum 1981, p. 149]. In other words, with respect to analysis model representations, this objective deals with the following questions:

- a. Can new analysis model instances be added?
- b. Can existing analysis model instances be changed?
- c. Can existing analysis model instances be deleted?

Current ad-hoc or specialized approaches to design-analysis integration suffer particularly in these respects. Unless a CAD/CAE tool was developed in-house, it is likely that the users have little, if any, means for adding or changing types of analysis models.

Another type of modularity, termed constructive modularity, also is desirable. Constructive modularity means that a representation of a particular analysis model can be built from predefined analytical building blocks (including representations of other analysis models). The newly represented analysis model can in turn be used as a component in the representation of another analysis model.

OBJECTIVE 7 Speed Give rapid results.

In order to be optimally useful during interactive design, a routine analysis model should provide results within seconds (or at most minutes) from the click of a mouse. The

execution (solution) phase of complex analysis models will often necessitate longer times, so this objective is largely limited to minimizing the time it takes for the creation (and interaction) of the analysis model(s).

OBJECTIVE 8 Flexibility Accommodate a wide variety of analysis models.

OBJECTIVES 20-29 define some of the axes along which analysis models vary. Therefore, this objective is to have those objectives achieved as a whole to a large degree.

OBJECTIVE 9 Directionality Allow different combinations of inputs/outputs.

Because there are different transformations that occur in engineering analysis, it is necessary to identify them and their natural and inverse directions. Using Figure 4.5 as a guide, these transformations are listed here and are explained in the indicated Objective.

- *Product-Analysis Transformations (PATs)* - See Objective 3 regarding product-analysis associativity.
- *Analysis Models* - See below in this Objective regarding the input/outputs of individual analysis models.
- *Analysis-Analysis Transformations (AATs)* - See Objectives 12 and 16 regarding interaction of multiple analysis models.

Directionality of Individual Analysis Models

As will be amplified in OBJECTIVE 20 (Purpose) a subset of the variables classified as inputs and a subset of the variables classified as outputs typically reverse roles in each of the above types of transformations when the analysis model is used for design synthesis versus design verification. Therefore, to the extent allowed by the nature of the

underlying relations, it is desirable to change which direction the analysis is run (i.e., which variables are outputs and which are inputs) to support such purposes.

The importance of this multidirectional capability is emphasized by the imminent premiere of a new journal, *Inverse Problems in Engineering* [GB, 1993]. The term *inverse* in the title typically means the output of *analysis* parameters that are normally considered inputs to analysis models. The term *multidirectional* is used instead in this thesis to include also the relations between analysis information and design information (PATs), and to emphasize that there is not just one possible output direction. According to the call for papers of this journal, inverse problems include those that determine shape, material properties, boundary values and initial values, forces, and even governing equations. With the exception of the last item, these items are typically **natural inputs** to analysis models that require solution techniques such as finite element analysis.

Table 4.4 Natural Inputs and Outputs of Finite Element Analysis

<u>Natural Inputs</u>	<u>Natural Outputs</u>
<ul style="list-style-type: none"> • system parameters • body /region parameters <ul style="list-style-type: none"> - geometry - material properties - loads - simplified boundary conditions 	<ul style="list-style-type: none"> • fields of state <ul style="list-style-type: none"> - stress - strain - deformation - temperature

Natural inputs to a finite element analysis (FEA) for thermomechanical problems are geometry, material properties, and simplified boundary conditions (e.g., forces, temperatures, and displacements in discrete or "nice" analytical forms such as uniform, linear, quadratic), and natural outputs include discretized fields for stress, strain, and displacement (see Table 4.4). This observation can be confirmed by reviewing FEA modeling languages such as Pre7 in ANSYS [SASI, 1990].

If the desired output is a natural input (e.g., a material property such as CTE), obtaining the solution typically will require an iterative method. Finite element programs often provide built-in procedures for such inverse problems. As an example, ANSYS offers "feature-based parametric optimization" functionality [SASI, 1993, p. 13] where natural inputs such as height, thickness, and number of rib stiffeners (for an ice cube tray, in their example) can be determined as outputs.

4.2 OTHER OBJECTIVES

OBJECTIVE 10 Final Results Provide final results.

The purpose of an analysis model typically is to give final results to some problem. This objective goes without saying, but is listed here as a comparison to the next objective.

OBJECTIVE 11 Intermediate Results Provide access to intermediate results.

Results determined at different steps in the analysis model can provide insight to the designer and can be used for judging the goodness of the results. Thus, access to their values is important.

OBJECTIVE 12 Interaction Enable interaction of multiple heterogeneous analysis models. As illustrated in Appendix B for the solder joint fatigue problem and in Figure 4.9, an analysis model can require several relatively smaller analysis models to exchange inputs and outputs in order to achieve the final results. The linkages that enable such interactions are termed analysis-analysis transformations (AAT) in Figure 4.5 and as noted in Objective 9 (Directionality). Supporting interaction of global/local analysis models (OBJECTIVE 16) is a special case of this objective with definite a natural and inverse directions.

Interacting analysis models may be different along any of the lines defined by OBJECTIVES 20-28. Niu and Shephard [1990] identify some of the issues in exchanging information between different types of finite element models (e.g., different mesh nodal points) and offer some solutions to such problems.

OBJECTIVE 13 Multivalued Inputs Allow multivalued input variables.

In some cases it would be helpful to supply a collection of values to input variables and get an collection of corresponding outputs as results (e.g., for parametric studies). This collection is not the same as an input vector in that each operation in the analysis should be performed on each member individually (e.g., scalar addition) in the case of a discrete collection. Also the collection could be a relation (e.g., a continuous interval).

OBJECTIVE 14 Multivalued Outputs Allow multivalued output variables.

This situation would occur not only when multivalued inputs are given (just described) but also when a relation involved in the analysis has multiple solutions (e.g., $x = \pm y$).

OBJECTIVE 15 Control Allow adjustment of solution procedure parameters.

Some solution procedures require parameters that would be otherwise independent of the actual analysis model if that procedure were not being used. Examples of such parameters which need control are mesh density and number of load steps for the case of the finite element solution method.

OBJECTIVE 16 Resolution Support hierarchical decomposition of regions (global/local models).

As previously mentioned this objective is to provide a special type of analysis model interaction capability (**OBJECTIVE 12 Interaction**) where a global analysis model (geometrically larger and coarse "grained") provides a local analysis model (geometrically smaller and fine grained) with boundary condition inputs. These linkages are AATs that perform in the natural direction in the manner just stated.

The intent of global/local analysis models is to keep the numerical computational size of each problem within reason yet get accurate results in the area of interest. The basic assumption in the global/local technique is that the global analysis model can provide reasonably accurate boundary conditions (e.g., displacements) around the local region of interest.

A recent exemplar application of this technique to electronic packaging (thermomechanical analysis of multi-chip modules or MCMs) is given by Shephard, et al. [1992] in their work that focuses on the interaction of global/local analysis models. Figure 4.10 illustrates the different levels of electronic packaging. Though this figure was intended to illustrate levels of physical and electrical interconnection, such levels are analogous to the regional levels that characterize global/local analysis models like those illustrated in Figure 4.9. Region 1 of that figure could correspond to the circuit pack (PWA) of the figure here.

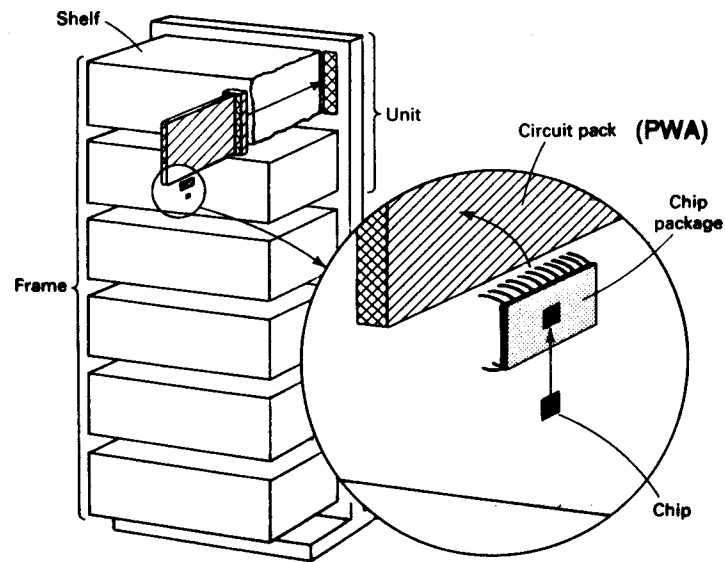


Figure 4.10 Typical Regions of Resolution in Electronic Packaging
[Pinnel and Knausenberger, 1989]

Note in Figure 4.10, too, for example that the PWA could be a local analysis model for a thermal analysis with the shelf or unit serving as the global analysis model. The PWA could then serve as the global thermal analysis model for the chip package (integrated circuit component) and so on in a hierarchical manner. Figure 4.11 shows the constraints between a PWA and its enclosure to emphasize the need to consider the next higher-level of assembly (here, the enclosure) when doing an analysis on the product of concern (the PWA). As can be seen, supporting the interaction of global/local analysis models is an important objective.

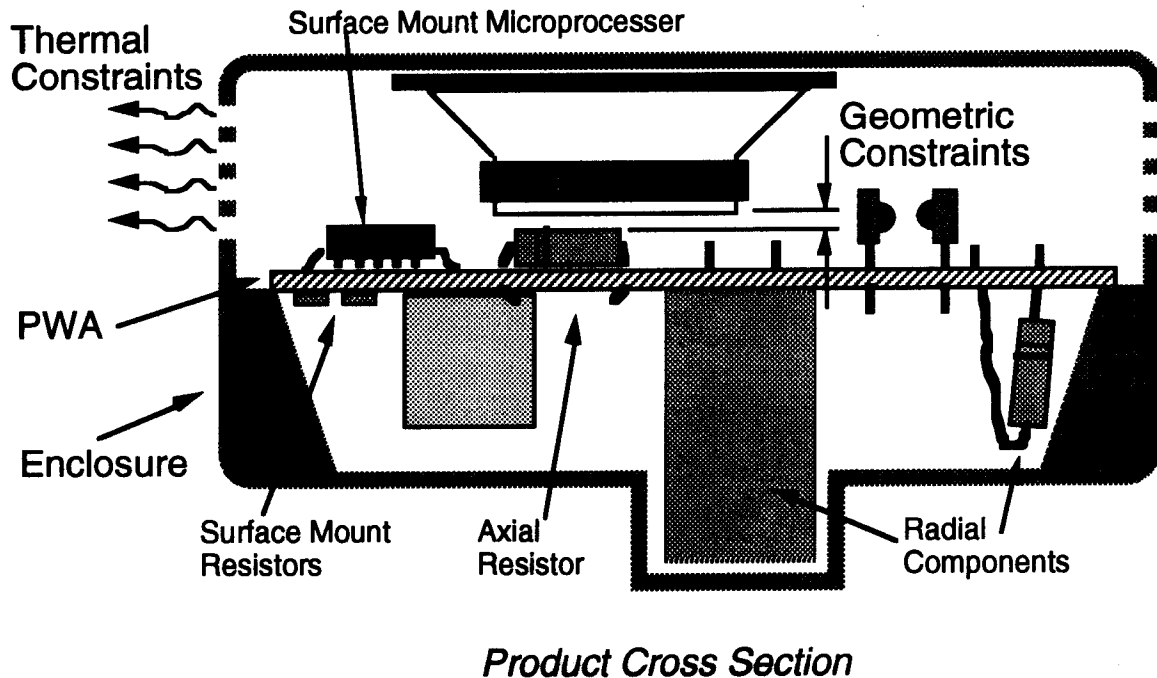


Figure 4.11 Constraints Between a PWA and its Enclosure

OBJECTIVE 17 Exchange Support analysis model exchange between CAD/CAE tools.

Unlike OBJECTIVE 12 Interaction, this objective is that the analysis model representation (PBAM) itself should be exchangeable between tools that would support the creation and usage of such representations.

OBJECTIVE 18 Compatibility Utilize existing CAD/CAE tools.

The representation should be able to utilize product and analysis data created by existing CAD/CAE tools if appropriate. It should also be able to use these tools to perform needed operations such as geometric transformations and analysis solution.

OBJECTIVE 19 Encapsulation Encapsulate existing tools that have specialized design-analysis linkages.

In some cases specialized tools exist that perform design and analysis in a tightly coupled fashion for limited types of products, applications, analysis model complexity. Therefore, rather than creating totally new representations of such analysis models, the proposed representation should utilize these capabilities by treating them as a black box to some degree.

OBJECTIVE 20 Purpose Fulfill different design/analysis purposes

Analysis models can be used for many different purposes during design, including:

Design Verification checking that the design meets some desired criteria.

Ex. Checking that the fatigue life of a solder joint on a PWA is acceptable for the intended design life and operation conditions.

Analysis Support Providing control-related input (OBJECTIVE 15) to another analysis model.

Ex. Estimating the initial load step for a nonlinear finite element analysis with the results from a simpler linear analysis model.

Design Synthesis Determining the value of a design attribute to meet some desired criteria.

Ex. Determine the height of a solder joint to meet a specified fatigue life. Note that now the height is an output and the fatigue life is an input which is the opposite of the case in the design verification example above.

Other purposes include optimization, sensitivity studies, analysis results verification, and published component behavior models (e.g., frequency varying resistance of a resistor).

When an analysis model is used for design synthesis purposes, the roles played by at least one input and one output variable reverse. For example, a typical design synthesis question would be "How thick should the wall be to keep the deflection less than 0.050"?, whereas the corresponding design verification question "Given that the wall thickness is 0.063", what will the deflection be?"

The significance of a multidirectional input/output capability was discussed in Objective 9 (Directionality). With respect to that discussion and Figure 4.5, design synthesis typically requires the inverse forms of PATs, analysis models (e.g., finite element-based models), and AATs. In contrast,

OBJECTIVE 21 Product Domain Be usable in multiple engineering product domains.

The basic structure of the representation should not limit its use to a single product domain. Example domains (also called industries) include electronics, automotive, aerospace, and marine.

OBJECTIVE 22 Product Type Support variation of product type within an analysis model.

Similarly, the representation should be able to support analysis models for many types and subtypes of products within a domain. Examples from the electronics domain include:

- Components: leadless/leaded; resistor/capacitor/integrated circuit, ceramic/plastic
- PWBs: single-sided/double-sided/multilayer; rigid/flexible/molded

The description of the case studies in Chapter 9 contains further examples of PWA product variations.

OBJECTIVE 23 Application Be usable in different applications for the same product domain.

Figures 4.3 and 4.12 illustrate how different analysis applications can exist for the same product type, and even within the same analysis discipline. In Figure 4.3 Application 1 might require a structural analysis of the product while Application 2 might dictate a thermal analysis or a structural analysis of another aspect of the product.

Hence, multiple analysis models of variable application need to be represented for a given product, especially for complex multidisciplinary products such as printed wiring assemblies. In general an analysis application could exist for each environment and situation a product encounters during its life cycle. For PWAs some example applications are solder joint fatigue, PWA warpage, plated through hole fatigue, and thermal reliability. One can see that the problems caused by the information gap in the current analysis modeling process are amplified greatly under these conditions.

OBJECTIVE 24 Discipline Represent analysis models from different engineering disciplines.

By definition it follows that multidisciplinary products, such as PWAs, can require numerous analysis models that are based on different engineering disciplines (Figure 4.12). Though there is definite commonality among many disciplines, the mathematical idiosyncrasies of any one discipline may be handled better by solution methods tailored to such differences, which leads to OBJECTIVE 26 (Solution Method).

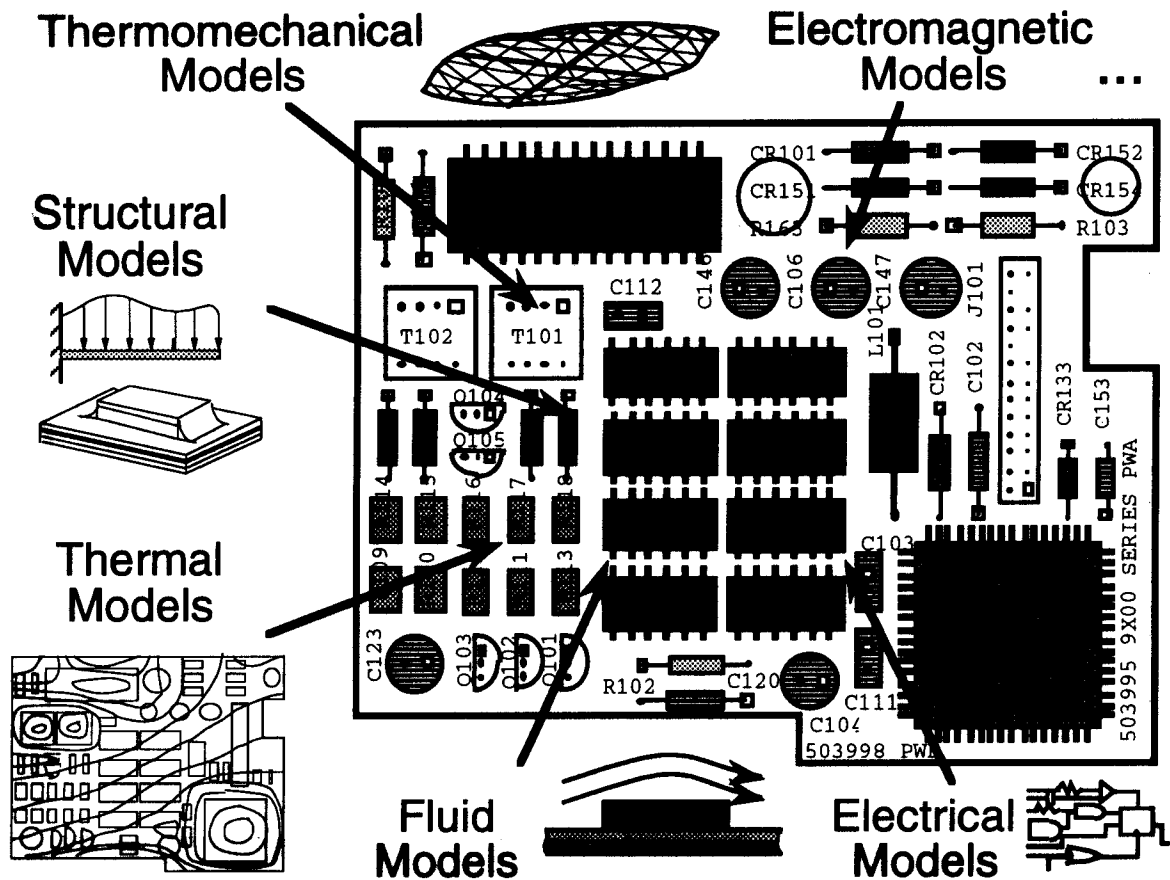


Figure 4.12 Analysis Models from Diverse Disciplines for Assorted Applications

OBJECTIVE 25 Behavior Regimes Support variation of behavior regimes within an analysis model.

It is not uncommon for a physical object to have distinct regimes of behaviors that depend on the range or magnitude of the same loading condition. For example, a beam can be modeled using linear deformation kinematic assumptions as long as the load on it produces deformations that are small compared to the size of the beam. However, even if all other parameters are the same, these assumptions will become invalid as the magnitude

of the load is increased. Thus, one could say that a beam has both linear and nonlinear regimes of deformation behavior.

Ideally an analysis model representation should contain the knowledge of what these regimes are, including their boundaries and conditions for crossing such boundaries.

OBJECTIVE 26 Solution Method Support different solution methods.

Again referring to Figure 4.3, note some of different forms of analysis model results:

- qualitative
- quantitative
 - exact
 - symbolic
 - numeric
 - approximate
 - numerically discrete

Each of these solution types has possibly several corresponding kinds of solution methods including numeric formula-based methods, symbolic methods, finite element methods, and finite difference methods.

OBJECTIVE 27 Variables Support different types of variables for analysis model input/output, and for interchange among analysis models.

Example types of variables common in engineering design and analysis include:

- product entities
- numerals
- symbols
- discrete collections:
 - arrays, vectors, matrices, strings, intervals
- relations

In this research is to provide quantitative variables. The reason for this focus is that the bulk of current engineering analysis models could be categorized as quantitative

analysis models, though qualitative forms are gaining popularity [Forbus in Bobrow, 1984; Kiriyaama, 1992].

OBJECTIVE 28 Relations Support different types of individual relations among variables.

By reviewing typical engineering textbooks [e.g., Crandall, et al., 1978] and industrial handbooks [e.g., Steinberg, 1988] one can see that relations typically encountered in engineering analysis include the following types:

Table 4.5 Some Categories of Analytical Relations

Kinematic relations	Conservation laws
Geometric properties	Boundary condition relations
Constitutive relations	Performance Definitions

Example mathematical forms of these relations include the following which can effect implementation capabilities.

- | | |
|-----------------------|--------------------|
| • discrete/continuous | • linear/nonlinear |
| • equality/inequality | etc. |
| • logical | |

As with variables, these relations have a somewhat natural correspondence with different analytical building blocks as discussed in Appendix F.

OBJECTIVE 29 Systems of Relations Support the solution of collections relations.

It is well known that, though an individual relation may be solvable by itself, often combining different types of relations together into a system of relations will dictate the use of different solution techniques, and may even have no solution. Types of systems encountered in engineering analysis models include:

- mixed relation types
- fixed/variable topology
- uncoupled/coupled
- linear/nonlinear systems
- steady state/transient
- simultaneous (cycles)
- eigenvalue problems
- boundary value problems

4.3 Summary

This chapter identified numerous capabilities that an analysis model representation ideally should enable, based on literature reviews, informal discussions, and personal observations. These capabilities were divided into THESIS OBJECTIVES (new capabilities that have received priority in this research) and OTHER OBJECTIVES (both existing and new capabilities that are desirable for any analysis model representation). These objectives might be viewed as a partial specification of the functionality desired in a highly advanced CAE system.

Given all the above objectives, the goal of this thesis is concisely stated in OBJECTIVE 1 (Representation) as repeated here:

Provide a single representation of routine analysis models which meets all other THESIS OBJECTIVES and many of the OTHER OBJECTIVES.

Having supported the worthiness of many of these objectives through the writings and thoughts of others, it is felt that providing a representation that achieves the preceding major objective will positively impact the theory and practice of engineering analysis in product design.

PART II ANALYSIS MODEL REPRESENTATIONS

CHAPTER 5

THE ANALYTICAL BUILDING BLOCK REPRESENTATION¹

*We cannot improve the language of any science
Without at the same time improving the science itself;
Neither can we, on the other hand, improve a science
Without improving the language or nomenclature which belongs to it.*
Antoine Lavoisier

This chapter introduces and defines entities that comprise analytical building blocks (ABBs). In particular, this chapter deals with generic, non-product-specific ABBs, i.e., ABBs that are not limited to a specific type of product. Chapter 6 defines the general PBAM representation that can be used to combine these building blocks with product model entities to form product-specific PBAMs. Chapters 7 and 8 describe how this approach of building a complex entity from other entities enables the modular development and implementation of PBAMs that represent specific analysis models.

First, a review of constraints and objects is given. Second, simple examples are used to show how constraints and objects can be combined to represent non-product-specific analytical building blocks. A graphical form of the resulting combination, called constraint schematics, is introduced.

5.1 Overview of Constraints

The basic idea of a **constraints** is that **variables**, a_i , and **relations**, r_j , among those variables can be declared explicitly by a user or application without explicitly specifying

¹ Condensed earlier versions of Chapters 5 and 6 have been published as [Peak and Fulton, 1993a] and [Fulton and Peak, 1993].

how any unknown variables are to be determined [Freeman-Benson, et al., 1990]. In a **constraint graph**, variables and relations are both vertices, and edges represent the participation of a variable in a relation. For example, consider the variables a_1, a_2, a_3, a_4 and the relations $a_1 + a_2 - a_3 = 0$ and $a_4 - (a_3 \cdot a_2) = 0$. The constraint graph of this case is given in Figure 5.1 where the convention of designating relations by boxes and variables by open circles has been used.

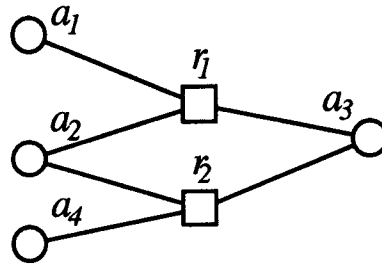


Figure 5.1 A Constraint Graph

A key advantage to viewing relations and variables as constraints is that constraints can be *multidirectional* (assuming such inversions are mathematically possible). In the above example if a_1 and a_2 are given as inputs, a_3 and a_4 will be determined. Likewise, a_2 and a_4 could be input to determine a_1 and a_3 . Constraint graph concepts will be formally defined in Section 5.1.4 after a brief review of the constraint literature.

5.1.1 Constraint Research and General Applications

Due to their "great expressive power" constraints have been the topic of much research in the past 30 years and have been widely used in graphics, engineering, and knowledge representation [Lassez, 1987]. Sutherland's Sketchpad system [1963], which dealt with geometric layout constraints, is referenced in almost every review as one of the earliest works involving constraints. Leler [1988] gives a highly readable introduction to what he

calls "constraint language programs" (i.e., an active constraint graph). Mackworth [1977] presents early work aimed at increasing the computational efficiency of constraint satisfaction problems (CSPs) by introducing algorithms that overcome the arc, node and path inconsistencies that plague backtracking algorithms (which tends to require solution times that are on the order of the number of variables involved).

Freeman-Benson, et al. [1990] define a "spectrum of algorithms" for solving constraint hierarchies which range from a more general purpose one (which can solve simultaneous equations) [Leler, 1988], to ones for linear equality/inequality constraint graphs, to their DeltaBlue algorithm. This last algorithm, which has been used in this research as described in Chapter 8, is designed for interactive applications (such as graphical user interfaces) and allows incremental changes to the constraint graph.

These authors also cite applications of constraints to five major areas: geometric layout, simulations, design and analysis problems, user interface design, and general purpose languages. Their later paper [Borning, et al., 1991] reviews such application of constraint hierarchies in particular. Constraint hierarchies are formed from constraints that have different strengths indicating the priority given to each constraint. This approach allows one to specify which variables will be inputs and which will be outputs as discussed in Chapter 8. Finally, Freeman-Benson and Borning [1992] describe their approach to integrating constraints (which are declarative) with objects (which are imperative). They note a basic strength to such an approach: the values of variables in constraints can be general objects. Thus, the domain over which constraints must be solved is not restricted to traditional mathematical domains (integers, reals, etc.), and the issues faced by constraint solvers become more plentiful.

5.1.2 Constraints Applications in Engineering

Representing engineering design relations and analysis relations as constraints is nothing new in general. Geometric modeling applications (e.g., parametric modeling) are perhaps the most prolific from both research and commercial product viewpoints [Roller, 1991; PTC, 1993]. Mantyla [1990] uses the DeltaBlue algorithm mentioned above to solve constraints representing geometric and assembly relationships in product design.

Philosophically, Sapossnek [1989] views the collective consideration of all relations in the product life-cycle as a constraint satisfaction problem. He distinguishes constraint-based design systems from some parametric systems in that the latter can require "causal ordering" of the relations for solution purposes. Serrano and Gossard [1988] demonstrate constraint management techniques in their Concept Modeler, a system for conceptual design. Domain-specific engineering constraint applications include PWB design for testability [Kim, et al., 1992] and copier paper handling system design [Mittal, et al. 1986].

Rinderle and Colburn [1990] identify the nature of design relations in general and those used during preliminary design in particular. Their figure of a constraint network DC motor under stall conditions is reproduced here because it graphically illustrates the complexity of even relatively simple engineering applications (Figure 5.2)

None of the papers reviewed discussed the linkages between detailed product and analysis models in terms of constraints. Also, as Figure 5.2 illustrates, analysis relations represented as constraints to date have typically been formula-based and are presented as "tangled knots" of constraints. No work was found that viewed the interactions between heterogeneous analysis models as constraints. This chapter introduces **constraint schematics** which are an attempt to fill some of these gaps.

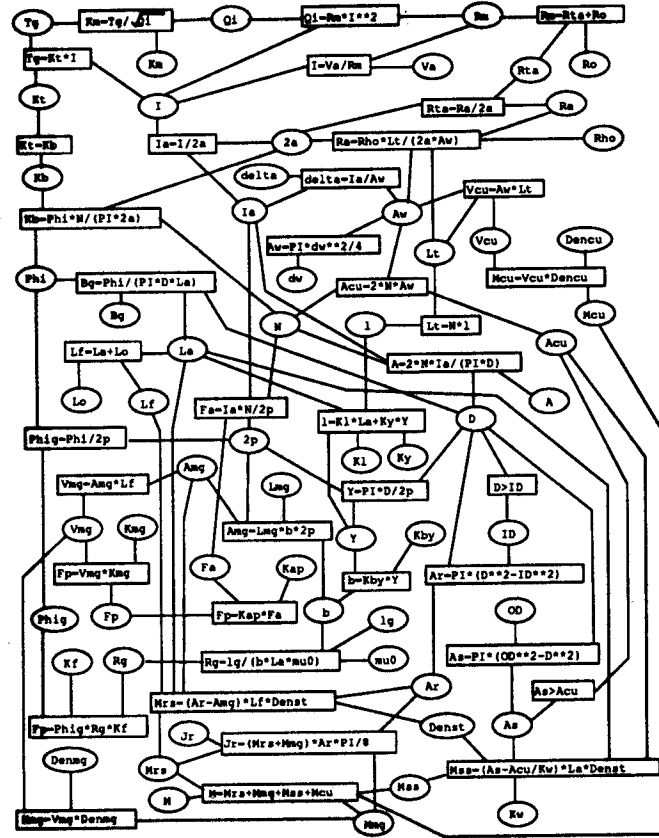


Figure 5.2 Constraint Network for a D.C. Electric Motor
[Rinderle and Colburn, 1990]

5.1.3 Constraint Graph Formalisms

A considerable theoretical foundation exists for constraint graphs as evidenced by the review included in Freeman-Benson, et al., [1990]. A few essential formalisms are now given as an introduction to this foundation, which in turn is based upon graph theory. As the new constraint schematic notation described later builds upon these formalisms, one can take advantage of the extensive existing theory.

DEFINITION 5.1 A **simple graph** consists of a set V , called the **vertices** of the graph, and of a set E , called the **edges** of the graph. E consists of 2-element subsets (unordered) of V denoted $P_2(V)$. A graph G can be represented by writing $G = (V, E)$ [after Bender and Williamson 1991, pg. 126].

DEFINITION 5.2 A **graph** is a triple $G = (V, E, \phi)$ where V and E are finite sets and ϕ is a function with range $P_2(V)$ and with domain E . E is called the set of **edges** of the graph G . The set V is called the set of **vertices** of G [Bender and Williamson 1991, pg. 127].

DEFINITION 5.3 A **variable** is a vertex that can participate in a constraint. A variable can have a value which is some arbitrary object [standard].

The symbols in Table 5.1 are defined so that constraint graphs can be graphically depicted (as in Figure 5.1) for enhanced comprehension. Symbol G1 denotes a variable.


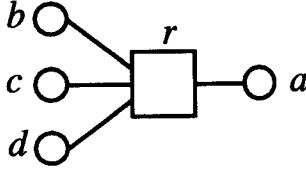

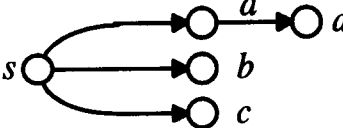
DEFINITION 5.4 A **constraint** expresses a desired relationship among one or more variables [Leler, 1988 pg. 6]. Formally, a constraint is an n -ary **relation** (Symbol G2) among a subset of variables, V . [Freeman-Benson, et al., 1990].

In the object-oriented programming implementation by Freeman-Benson, et al., [1990] each constraint has a set of methods, any of which can be executed to cause the constraint to be satisfied. Each method uses some of the constraint's variables as inputs and computes the remainder as outputs. A method may only be executed when all inputs and none of its outputs have been determined by other constraints.

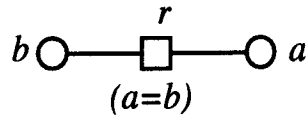
DEFINITION 5.5 A **constraint graph** is a graph consisting of a set of variables, V , and a set of constraints (n -ary relations), C , where each variable in each constraint in C is a member of V [standard].

Figure 5.1 is an example constraint graph that has been graphically depicted using variable symbols (G1) and relation symbols (G2), which are taken from the work of Maloney [1991, p. 33].

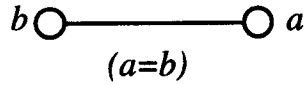
Table 5.1 Extended Constraint Graph Notation

	Graphical Symbol	Meaning	Equivalent Text
G1	Variable 	a is a variable .	a
G2	Relation 	Variables a , b , c , and d are related by relation r . This relation can be written as $r(a,b,c,d)$.	$r(a,b,c,d)$
G3	Equality Relation 	Variables a and b are equal, i.e., an equality relation exists.	$a=b$
G4	Part-of Relation 	Variable s has attributes a , b , and c which are variables (i.e., they are part-of s). Variable d is an attribute of a and a subattribute or subvariable of s . I.e., variables s , $s.a$, $s.b$, $s.c$, and $s.a.d$ are shown.	Dot Form $s.a.d$ $s.b$ $s.c$ Indented Form s a d b c

Due to its common use and importance, a new graphical symbol (G3) has been defined in this research for perhaps the simplest relation, the **equality relation**. The equality relation is a special binary relation with the literal depiction given in Figure 5.3 (a.) using the general relation symbol (G2). In this figure, the relation $r(a,b)$ has two forms: $a=b$ and $b=a$. To make constraint graphs less cluttered, this special relation is shown graphically using the abbreviated Symbol G3 as shown in Figure 5.3 (b.).



a. Literal Notation



b. Specialized Notation

Figure 5.3 Equality Relation Notations

Another minor development original to this research is the semantic extension of an existing notation, the **part-of relation** (Symbol G4). This symbol and related meaning is based on the data decomposition symbol by Rumbaugh, et al [1991, p. 126]. They defined their notation with apparently no intent towards use within constraint graphs. The textual form of the part-of relation can be expressed using dots (.) to delimit the parent-child part-of relation, where the parent object is shown to the left of the child object. An equivalent textual indented form is shown in Table 5.1 where the child indented to the right on a line below the is parent. In either form, the object that is at the top of the part-of hierarchy is the left-most object and is called the **eldest** object.

Considering the changes made by differentiating equality relations and part-of relations from other relations, a new type of constraint graph is now defined:

DEFINITION 5.6 An **extended constraint graph** is a constraint graph in which equality relations and part-of relations are distinguished from other types of relations.

Due to the two new symbols (G3 and G4), the symbols given in Table 5.1 collectively form what is called **extended constraint graph notation**. An extended constraint graph can be shown graphically using extended constraint graph notation.

For the purpose of example, part-of relations (i.e., relations r_{11} through r_{17}) and associated variables have been added to Figure 5.1 to produce the exemplar constraint graph in Figure 5.4. This example uses only "standard" constraint graph notation (i.e., using only Symbols G1 and G2) so it is difficult to tell which relations are part-of relations. Figure 5.5 is an example *extended* constraint graph of Figure 5.4. Note all the part-of relationships are indicated by the small filled triangles (Symbol G4) in Figure 5.5. (Since this figure is intentionally generic, there are no "natural" semantics behind these part-of relationships. The significance of such relationships will be illustrated later with specific examples.) The variable s is the object that encapsulates all the variables and relations shown, and is, therefore, the eldest object with respect to the part-of relation. In summary, these generic examples show how an extended constraint graph combines the part-of relation from the object representation with general relations from constraint graphs.

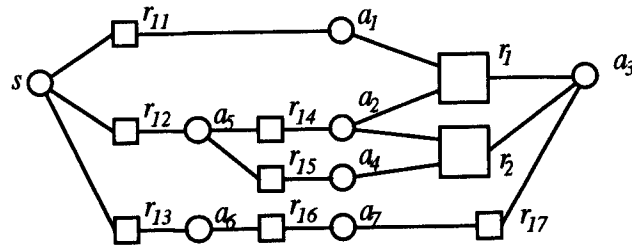


Figure 5.4 Example Constraint Graph

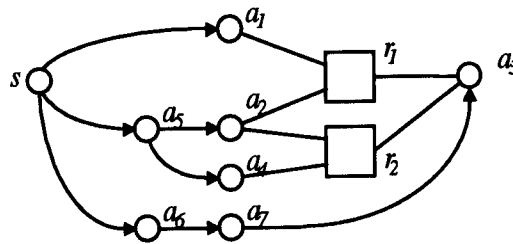


Figure 5.5 Example Extended Constraint Graph

5.2 Overview of Analysis Model Object Representations

The advantages of the object representation for engineering applications have been widely proclaimed [Fenves, 1990; Lee and Arora, 1991; Filho and Devloo, 1991; Whelan, 1989] and include the following:

- Close relation to the engineering view of entities.
- More natural representation of complex engineering objects and the part-of relationship.
- Association of specialized behavior with an object.
- Inheritance of attributes and behavior through the is-a relationship.

Dym and Levitt [1991] declare the object representation to be an important case of declarative programming. Salustri and Venter [1992] develop axioms, definitions, and theorems based on set theory which formally describe notions of objects, object equivalence, facets of attributes, and specialization, etc. with the intent of using objects to

capture design information. Lee and Arora [1991] and Filho and Devloo [1991] give simple engineering application examples in which a matrix class is defined. Fenves [1990] includes an example on finite element node ordering.

Appendix A gives an overview of object concepts in general. Examples of how objects or object-like entities have been used specifically to represent analysis concepts include:

- Aerodynamic entities [Stephens, 1993]
- Control volume, contact (rolling contact, ball-in-socket, etc.), mass, rigid link, etc. [Mashburn and Anderson, 1991]
- Column, beam [Jain, et al., 1991]
- 2D truss [Miller, 1988]
- STEP Materials [ISO 10303-45]
- STEP Kinematics [ISO 10303-105]

None of these examples include derivational knowledge or analysis model assumptions/limitations as part of the representation. Most disturbing is the fact that, with the exception of Stephens, Mashburn & Anderson, and STEP, *no hierarchies* of analysis entities are shown. This lack is a very significant shortcoming in that the is-a hierarchy is one of the most fundamental and powerful concepts in the object representation. Mashburn and Anderson's hierarchy of contacts (for use in rigid body dynamics, it appears) looks promising, but their partial hierarchy of general analysis models appears to be inconsistently categorized.

The development of a good hierarchy of analysis entities is probably one of the more challenging aspects of applying the object representation to engineering analysis. It is not clear that only one "good" hierarchy exists as evidenced by the debate over multiple inheritance [Salustri and Venter, 1992].

5.3 Constraint Schematics of Analysis Models

This section shows how constraints and objects can be combined to represent analysis models from a non-product-specific viewpoint. With this intent in mind, a new notation, termed **constraint schematics**, has been developed in this research to depict constraint graphs of engineering analysis models (a formal definition of a constraint schematic is given later in this section).

It will be shown in this section that an analysis model (from a constraint viewpoint) can be represented as a constraint graph with a structure much like that encountered in electrical schematics. Therefore, for familiarity purposes analogous electrical schematic terminology and symbols have been used to define the constraint schematic notation where possible. All the graphical symbols in this section are results of this research (except for duplicated extended constraint graph symbols from the preceding section).

Table 5.2 introduces some basic constraint schematic symbols. Symbols marked with an asterisk (*) have the same meaning as corresponding symbols in the extended constraint graph notation given in Table 5.1. The variable symbols (S1 and G1) are exactly the same, as are the equality relation symbols (S3 and G2); they are repeated here for convenience. For the sake of convention and style, the graphics of the relation symbols (G2 and S2) and the part-of relation symbols (G4 and S5) differ slightly. Every connection between variables and relations in a constraint graph is a separate straight line or curve per mathematical convention. Connections in a constraint schematic run vertically and horizontally and can contain equality junctions (Symbol S4) per electrical schematic convention.

Table 5.2 Basic Constraint Schematic Notation


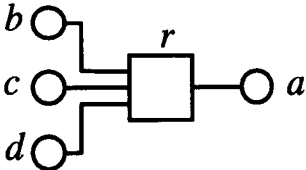

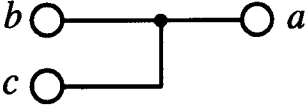
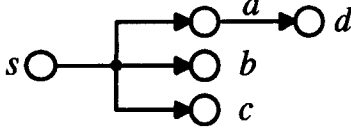
	Graphical Symbol	Meaning	Equivalent Text
*S1	Variable 	a is a variable .	a
*S2	Relation 	Variables a , b , c , and d are related by relation r . This relation can be written as $r(a,b,c,d)$.	$r(a,b,c,d)$
*S3	Equality Relation 	Variables a and b are equal, i.e., an equality relation exists between them. (See Note 1)	$a=b$
S4	Equality Junction 	Variables a , b , and c are equal.	$a=b=c$
*S5	Part-of Relation 	Variable s has attributes a , b , and c which are variables (i.e., they are <i>part-of</i> s). Variable d is an attribute of a and a subattribute or subvariable of s . I.e., variables s , $s.a$, $s.b$, $s.c$, and $s.a.d$ are shown.	Dot Form $s.a.d$ $s.b$ $s.c$ Indented Form s $\quad a$ $\quad\quad d$ $\quad b$ $\quad c$

Figure 5.6 is the constraint schematic form of the constraint graph given in Figures 5.4 and 5.5. Admittedly, at this point the differences between the extended constraint graph (Figure 5.5) and this constraint schematic are superficial. Additional notation to be introduced in this section will manifest the important differences between the two notations.

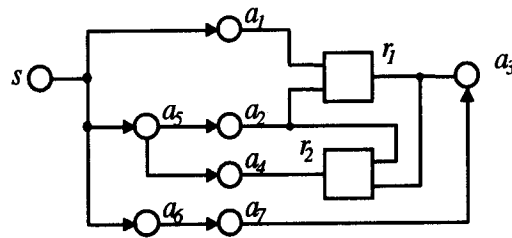


Figure 5.6 Verbose Constraint Schematic of Object s

When the eldest object in the constraint schematic (or extended constraint graph) contains all the variables and relations in the constraint schematic, the part-of relations involving the eldest object need not be shown. In Figure 5.6 since s meets this criteria, the constraint schematic shown is "verbose". The more commonly used shorthand notation (without s) of this same constraint schematic is illustrated in Figure 5.7. Similarly, the eldest object need not be shown in either of the textual forms of the part-of relation hierarchy (the dot form or the indented form). Such shorthand notation will be used from here on out except where the longer form is needed for illustration purposes.

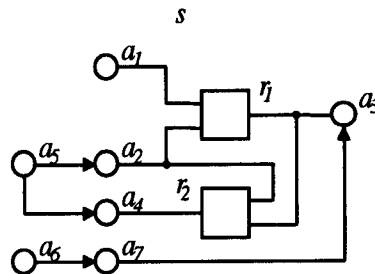

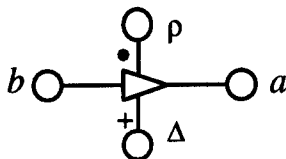
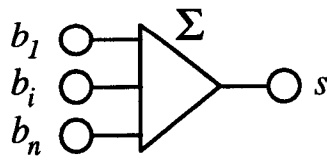
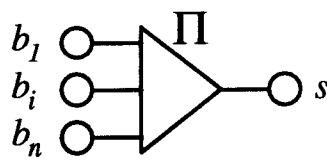


Figure 5.7 Constraint Schematic of Object s

As some relations occur quite frequently in engineering analysis, it is convenient to introduce specialized symbols for these mathematical relations (Table 5.3). The symbol used for the **absolute value relation** (M1) is based on the electrical schematic notation for a diode because they perform analogous functions in one direction: they do not let negative values pass through. The symbol used for the **scale & offset relation** (M2) comes from operational amplifiers that scale electrical signals by a specified gain.

Table 5.3 Mathematical Constraint Schematic Notation

	Graphical Symbol	Meaning	Equivalent Text
M1	Absolute Value 	An absolute value relation exists between a and b .	$a = b $ $b = \pm a$
M2	Scale & Offset 	A scale & offset relation exists between a and b . If not shown, the value of ρ (the scale ratio) defaults to 1, and Δ (the offset delta) defaults to 0.	$a = \rho \cdot b + \Delta$, $b = (a - \Delta) / \rho$
M3	Summation 	A summation relation exists between s and b_i .	$s = \sum_{i=1}^n b_i$
M4	Product 	A product relation exists between s and b_i .	$s = \prod_{i=1}^n b_i$

Though a few such symbols are given here, it is obvious that many more could be developed. However, it is not desirable to have too many symbols or the ease of recognition intended by the graphical notation will be diminished.

EXAMPLE 5.1 Analytical Primitive (Elementary Rod)

To solidify the above concepts and show how constraint schematics can be used for analysis applications, a simple example will be introduced which will be built upon later. Figure 5.8 is an analysis model, a simple rod, which can elongate due to an axial load, F , and/or a thermal load, ΔT .

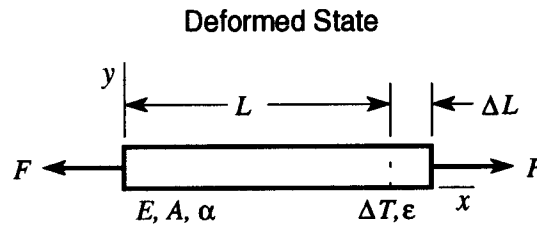


Figure 5.8 A Simple Rod

The relation between strain, ϵ , total elongation, ΔL , and these loads is given by the following equations, where T_o is the reference temperature and T is the current temperature of the rod. (These equations can be found in or derived from any elementary mechanics of materials text. See, for example, the text by Crandall, et al., [1978].) L is the length of the rod in the undeformed state (when $F=0$ and $T=T_o$), and A is the cross sectional area of the rod. E and α are the material properties Young's modulus and coefficient of thermal expansion (CTE) respectively.

$$\Delta T = T - T_o \quad (5.1)$$

$$\epsilon = \frac{F}{EA} + \alpha \Delta T \quad (5.2)$$

$$\Delta L = \epsilon L \quad (5.3)$$

As this example is intended for instructional purposes, other possible load conditions and variations have been neglected. Also, the way this example divides these relations and associated variables into objects neglects the origin of these relations to some degree for the sake of simplicity. Hence, such example objects will include the word **Elementary** in their name to distinguish them from any realistic counterparts that are used in actual case studies. The solder joint fatigue case studies discussed later demonstrate how more realistic analysis models with greater complexity can be represented using constraint schematics.

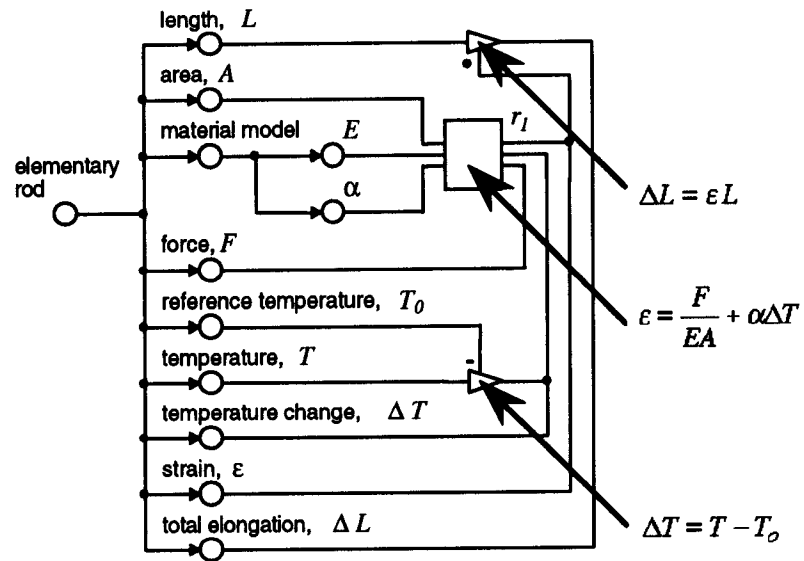


Figure 5.9 Verbose Constraint Schematic for Elementary Rod

The above analysis model can be represented as an analytical primitive having the following constraint schematic (Figure 5.9). An **Elementary Rod** is an object with the following attributes as indicated by the part-of relationship: L , A , F , T_0 , T , ΔT , ϵ , ΔL . Since variables E and α are properties for a linear-elastic material, they are logically grouped together as part of another object which the **Elementary Rod** calls a **material model**

(thus, they are subattributes of the Elementary Rod). Eqns. 5.1 and 5.3 are represented in the constraint schematic as scale and offset relations. Eqn. 5.2 is represented by the relation r_I which shows connections to its associated variables.

Figure 5.10 shows this same constraint schematic in the normal shorthand form (without the eldest object part-of relations) where it is understood that the Elementary Rod object contains all the variables and attributes relations shown (either directly, as attributes, or indirectly, as subattributes).

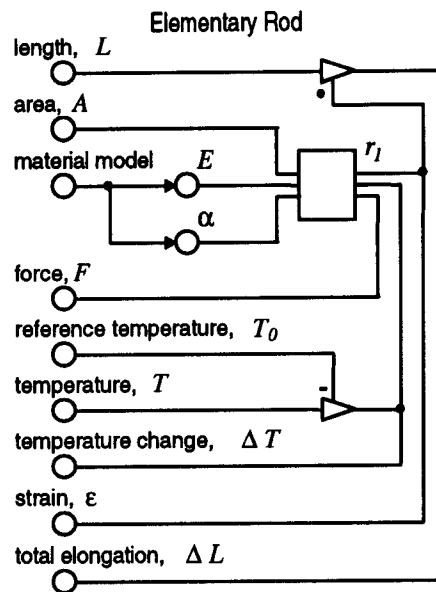


Figure 5.10 Constraint Schematic for Elementary Rod

This section has introduced a new notation, constraint schematics, which graphically shows the relations and variables in analysis models. More constraint schematic notation will be defined later in this chapter. To give the proper perspective up front, it is important to note that a constraint schematic is just one **view** of an analysis model, that is, it is a *subset* of the complete information model that represents an analysis

model. Other analysis model views will also be defined in this chapter, including the **ABB structure** which is considered the *master view*, as all other views can be derived from it.

5.4 OBJECT RELATIONSHIP DIAGRAMS

An **object relationship diagram** is one other view of an ABB. It is a partial graphical view of the overall ABB analysis model representation (information model) that emphasizes the is-a and part-of relations between objects used to represent analysis models. Numerous notations exist that could be used for this purpose, including NIAM [Nijssen and Halpin, 1989], IDEF1X [IDEF1X, 1985; Bravoco and Yadav, 1985c], entity-relationship (ER) diagrams [Chen, 1979], and the Object Modeling Technique (OMT) Object Model notation [Rumbaugh, et al., 1991]. Chadha, et al. [1991] give an appraisal of these methods for engineering applications. EXPRESS-G [ISO 10303-11] has been used in this research as the notation for object relationship diagrams, since it conveys the is-a and part-of relationships in a straight forward manner and is also an ISO standard. See Appendix A for a review of the basics of the EXPRESS-G notation

An object relationship diagram of the analysis model in Example 5.1 is depicted in Figure 5.11 (along with a summary of the EXPRESS-G notation for convenience). This diagram corresponds with the constraint schematic given in Figure 5.9 (and Figure 5.10). The is-a relationship (indicated by a bold line) shows that an **Elementary Rod** is a special type of **Elementary Deformable Body**. As all deformable bodies have a reference temperature and a linear-elastic material model, these attributes are part of the **Elementary Deformable Body** object. In this way such common attributes can be inherited by **Elementary Rod** and other subclasses such as **Elementary Beam**.

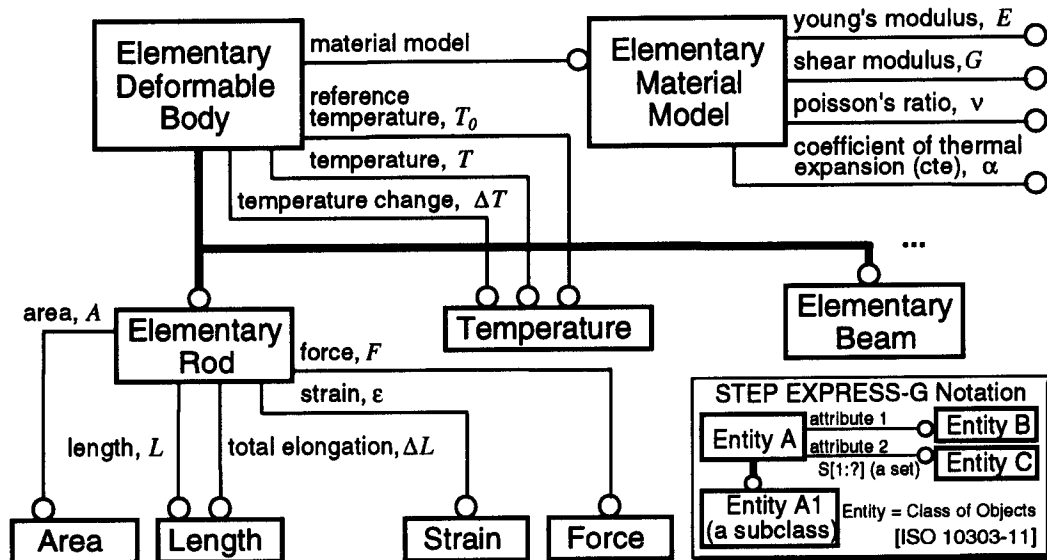
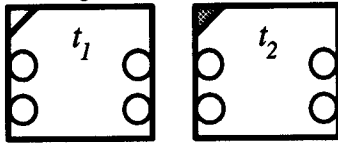


Figure 5.11 Object Relationship Diagram for Elementary Rod

Note that the attributes are not just simple numbers (e.g., reference temperature is of type **Temperature**). Instead, they are objects themselves which can have their own attributes, such as units (not shown). The material model attribute of **Elementary Deformable Body** is of type **Elementary Material Model**. This object itself contains other variables (e.g., ν and G) and relations which are not needed in this context and, therefore, are not shown in the constraint schematic in Figure 5.9.

As demonstrated in this example, the object relationship diagram conveys which objects are specializations of other objects (the is-a relationship - bold lines), as well as which objects are building blocks in other objects (the part-of relationship - thin lines). Thus, an object relationship diagram shows which *variables* are inherited from superclass objects; however, it does not show which *relations* are inherited. Therefore, the inheritance notation given in Table 5.4 depicts both variable and relation inheritance information. (Subsystems are explained in the next section).

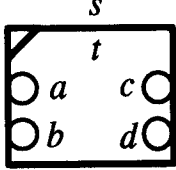
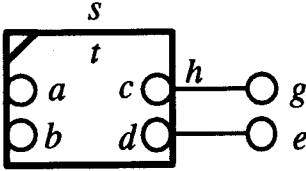
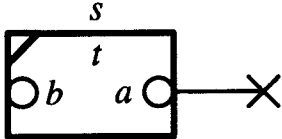
Table 5.4 Constraint Schematic Inheritance Notation

	Graphical Symbol	Meaning	Equivalent Text
S9 ²	<p>Inheritance</p> <p> a_1 ○ a_2 ● r_1 □ r_2 ■ s_1 s_2 </p> 	<p>Unshaded variable a_1, relation r_1 and subsystem s_1 are inherited from the super class.</p> <p>Shaded variable a_2, relation r_2, and subsystem s_2 and related connections are new to the class containing this constraint schematic.</p>	

For example, Figure 5.12 is the constraint schematic for Elementary Rod (Figure 5.10) that has been annotated with shading to show inheritance information. Since the Elementary Rod class inherits temperatures T_o , T , and ΔT from the Elementary Deformable Body class, these variables are shaded. Such inherited variables (as well as subsystems and relations) need to be specified only in the analytical building block superclass, while new variables (and subsystems and relations) are specified in the new analytical building block class. For example, area, A , is a variable that is new to the Elementary Rod class. The scale and offset relation between T_o , T , and ΔT is inherited from the superclass (Elementary Deformable Body). In summary, shaded variables, relations, and subsystems in a constraint schematic indicate that they are inherited from a superclass object.

² The constraint schematic symbols are not presented consecutively in this chapter. Their numbering is based on their logical order as a whole set as summarized in Appendix C.

Table 5.5 Basic Constraint Schematic Subsystem Notation

	Graphical Symbol	Meaning	Equivalent Text
S6	<p>Subsystem</p> 	Variable s is a subsystem of type t which has attributes and/or sub-attributes a through d .	
S7	<p>Semantic Linkage</p> 	Variable $s.c$ is known as h in the scope outside of t by a semantic linkage (i.e., $h=s.c$ and $g=h$). Variables a , b , and d are semantically linked with the same names in both scopes (i.e., $a=s.a$, $b=s.b$, $d=s.d$, and $d=e$).	$a=s.a$ $b=s.b$ $h=s.c$ $d=s.d$ $g=h$ $d=e$
S8	<p>Invalid Variable</p> 	Variable a is not valid in the scope outside of t .	a is <i>invalid</i>

appropriate if the user is only interested in relation r_1 and its associated variables, a_1 , a_2 , and a_3 .

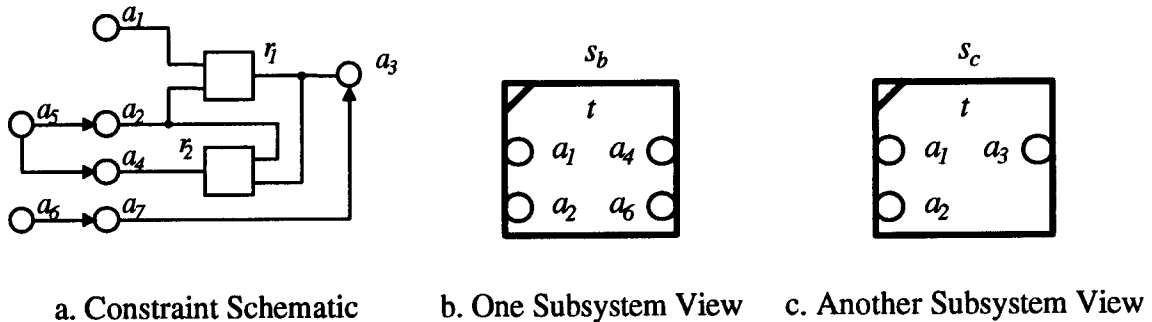


Figure 5.13 Some Views of an Object

Note that the full constraint schematic is present and active in a subsystem no matter which variables are shown on the subsystem symbol; this view is simply an abstraction that hides detail which is unnecessary to the subsystem user. This concept is analogous to an electrical integrated circuit (IC) pinout diagram which does not show the detailed circuitry inside the IC. Thus, the subsystem symbol (S6 in Table 5.5) is based on IC pinout diagrams and is one realization of the "software IC" concept.

The subsystem view is one primary new aspect of constraint schematics that is brought about by merging the constraint and object concepts of relations, information hiding, and encapsulation. Just as an integrated circuit can be a component in a larger electrical circuit, an object can be a subsystem in the constraint schematic of another object. Since constraint schematics can contain arbitrarily deep subsystem nestings (unlike physical ICs), subsystems provide a way to organize the previously mentioned "tangled knots" of constraints (Section 5.1) into meaningful, more comprehensible bundles. The following definition and properties summarize this discussion:

DEFINITION 5.7 A **subsystem** is an analytical building block that is viewed by a subset of its variables.

PROPERTIES

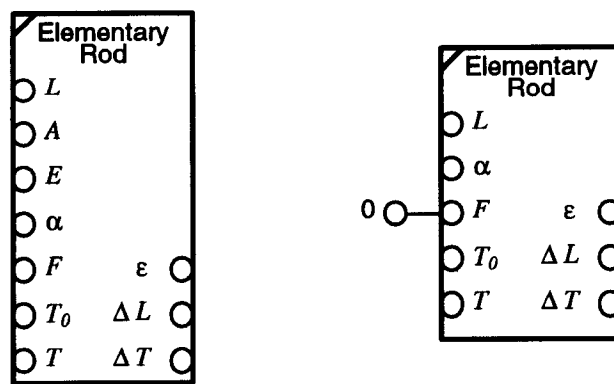
1. A subsystem can be used in the constraint schematic of another object.
2. Subsystems can be nested arbitrarily deep, i.e., a subsystem can contain other subsystems that in turn contain other subsystems, and so on.

Now that this new concept, subsystems, has been introduced, a definition of a constraint schematic can be given as follows:

DEFINITION 5.8 A **constraint schematic** is the union of one or more extended constraint graphs whose variables and relations have been classified according to the constructs given in Table C.2.

One major difference between a constraint schematic and an extended constraint graph is that collections of variables and relations may appear in a constraint schematic as arbitrarily deep nested subsystems. Another major difference, options, will be discussed later in this chapter.

Continuing with the analysis model from Example 5.1, two possible Elementary Rod subsystem views are given in Figure 5.14. The first view is a "complete" subsystem in that it shows all the variables involved in the analysis model relations. In the special case where $F=0$, the second view shows only the variables that are needed to obtain ΔL , ΔT , or ϵ (natural outputs for this problem). Thus, if one is only interested in the attributes related to ΔL that deal with material, geometry, and thermal loads (natural inputs for this problem), then the second subsystem will suffice.



a. Complete Subsystem b. Special Case Subsystem

Figure 5.14 Example Subsystem Views for Elementary Rod

By convention in this thesis, variables that are natural inputs generally are placed on the left of the subsystem symbol, while natural outputs are on the right. Furthermore, if there are more than one body, variables tend to be grouped together according to which body they are part of, and variables associated with several bodies (e.g. reference temperature or other system variables) are at the top of the subsystem symbol. These conventions, however, are not always strictly followed (and need not be).

EXAMPLE 5.2 Multibody Analysis Model (Interconnected Rods System)

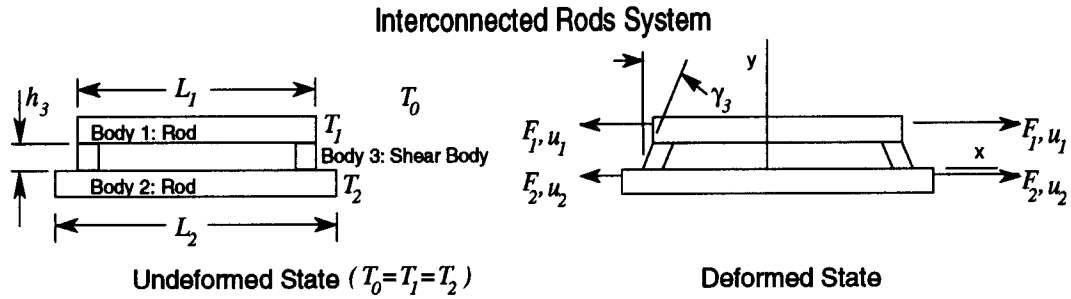


Figure 5.15 An Exemplar Multibody Analysis Model

As another example, consider the analysis model in Figure 5.15 which has multiple bodies. One representative use of this generic analysis model is given later in the case study PBAM of Engelmaier's extensional model (Chapter 9). It contains the following relations that determine the thermal expansion mismatch, $\Delta(\alpha\Delta T)$, and calculate the shear strain in body 3, γ_3 . Note that here the subscripts refer to generic body numbers and not to the physical entities being modeled in a specific product.

$$\gamma_3 = \frac{L_1 \Delta(\alpha\Delta T)}{2h_3} \quad (5.4)$$

$$\Delta(\alpha\Delta T) = \alpha_2(T_2 - T_0) - \alpha_1(T_1 - T_0) \quad (5.5)$$

Interconnected Rods System (IRS) An analytical system composed of two Rods and two Shear Bodies has been created to represent this analysis model. The partial constraint schematic of this system, and one possible subsystem view, are given in Figure 5.16 (for the case where $F_1 = F_2 = 0$). Only one Shear Body is used in this schematic due to symmetry. A more complete description of this system is given in Appendix F.

Note that in this example, Eqn. 5.5 is represented explicitly in the analytical system. This relation probably could be achieved better by connecting the appropriate boundary condition variables between the bodies; however, such an approach has not been pursued here. The end result is that the bodies in this case serve primarily as data structures, as their inside relations are not utilized.

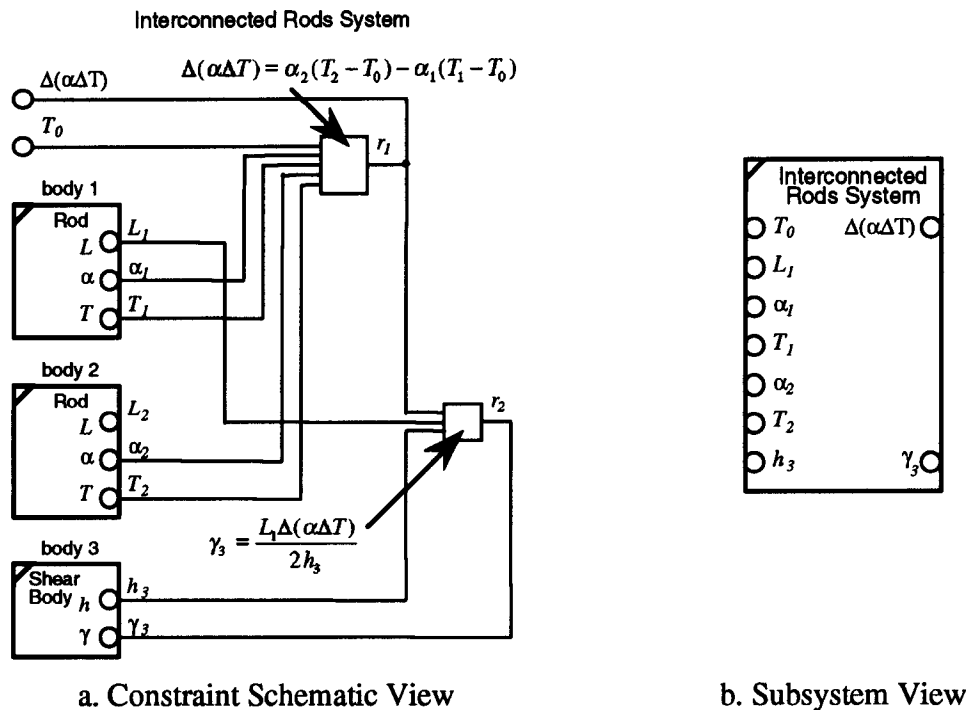


Figure 5.16 Two Views of an Analytical System

This example illustrates the concept of the **scope** of variables (see EXPRESS [ISO 10303-11] for a similar discussion on this concept). The scope of a variable is the context within which its name is valid (i.e., the variable can be accessed simply by its name within its scope). Thus, a variable can be known by one name within a subsystem and by another name outside the subsystem. For example, the height of body 3 is known simply as **height**, *h*, with respect to the **Shear Body** subsystem. (i.e., it is known as *h* in the scope of this subsystem) However, with respect to an **Interconnected Rods System** (the subsystem's usage scope), the same height is specifically known as the height of body 3, *h₃*, (that is, **body3.height**). Per Symbol S7 (Table 5.5), such **semantic linkages** are indicated by placing the name of the variable with respect to the usage scope beside the variable in the subsystem. A semantic linkage assigns a name to a variable that is meaningful within the usage scope.

The **invalid variable** symbol (S8) in the same table can prove useful when variables in a subsystem are not valid with respect to the user of the subsystem. This situation can occur, for example, when the user knows that some subsystem assumptions have been violated such that some variables are still useful while others are not.

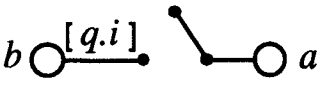
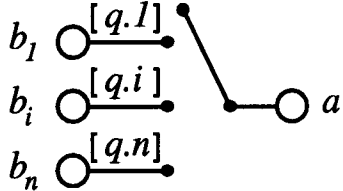
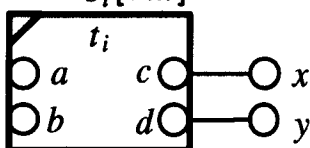
5.6 Analysis Model Options

To enhance the flexibility of analytical building blocks, the ability to have different **options** is included in their representation. Options allow the analytical building block user (either a person or an application) to specify variations in analysis model operation. OBJECTIVE 5 (Options) in Chapter 4 explained the usefulness of analysis model options and gave examples, including complexity variations and inclusion of additional effects. To illustrate this concept further, the rod model in Example 5.1 could have an **option category** called **material model** with possible options **Linear Material Model** and **Nonlinear Material Model**.

It is important to note that options may or may not be independent of the values assigned to the variables in the analysis model. In some cases the assumptions of an analysis model may dictate which options should be chosen. For example, in Example 5.1 the rod model assumes that the material behaves linearly. If a given load causes a stress in the rod that is greater than the yield stress, the nonlinear material model option should probably be used. Depending on the purpose of the analysis model and how much greater the calculated stress is than the yield stress, one may still opt to use the linear material model. Such analysis modeling decisions should be based on engineering judgment for the problem at hand. Therefore, as it is sometimes debatable if an option is independent of the given analysis model conditions, in this thesis *it is assumed either the ABB user specifies each option, or default options are used.*

Drawing on electrical switch terminology, in constraint schematic notation, an option category is a special kind of **constraint switch** that is labeled with option numbers enclosed in brackets (Symbols S10 and S11 in Table 5.6). Options also can be written in textual form as indicated in the table. These extended switch concepts and associated symbols are believed to be new to this research, though Leler briefly mentions simple switches as a type of **higher order constraint** [1988, p. 136]. More definitions are needed at this point to root these concepts into constraint graph theory (in brief, it will be established that options can be represented as subgraphs in a constraint graph).

Table 5.6 Constraint Schematic Switch/Option Notation

S10	<p>Two Position Switch</p> 	<p>A two-position pole equates a and b when switch position i of switch q is selected ($i=1,2$). In other words, $a=b$ when option i of option category q is chosen.</p>	$a=b [q.i]$
S11	<p>Multi-Position Switch</p> 	<p>An n-position pole equates a and b_i when position i of switch q is selected (i.e., when option i of option category q is chosen), for $i=1...n$.</p>	$a=b_1 [q.1]$ $a=b_i [q.i]$ $a=b_n [q.n]$
S12	<p>Subsystem Substitution</p> 	<p>Switch m contains an n-position pole between each connected variable pair. (e.g., $s_i.c=x$ when position i is selected for $i=1...n$)</p> <p>In an ABB, option category m indicates sub-system s can be one of n possible types of objects, $s_i=t_i$, for $i=1...n$.</p>	$s_i.c=x [m.i]$ $s_i.d=y [m.i]$ for $i=1...n$

DEFINITION 5.9 A **higher order constraint** is a constraint on constraints. A **second order constraint** can be treated as a combination of first-order constraints. An example of such a constraint follows [Leler, 1988, pg. 32]:

$$\text{if } (x=y) \text{ then } (b=c/a)$$

Note that this constraint can change the topology of the constraint graph to which it belongs. If the constraint $(x=y)$ is met, the constraint $(b=c/a)$ will be created between variables a , b , and c .

DEFINITION 5.10 A **constraint subgraph**, S , is a constraint graph which is intended to be connected to another constraint graph, G [standard].

PROPERTIES

1. A constraint subgraph, S , is **connected** to a constraint graph, G , if at least one variable in F is also a variable in G .
2. A constraint subgraph, S , contains references to the variables that it can share with G . S will actually share those variables only if it becomes connected to G .
3. A constraint subgraph is the **primary** constraint subgraph in a constraint graph if it is the subgraph to which all other constraint subgraphs are connected.
4. A constraint subgraph is **active** if it is connected to another constraint graph.

DEFINITION 5.11 A **constraint switch** is a higher order constraint which has a finite number of **switch positions** and a **common constraint subgraph**, S_0 . Each switch position is a constraint subgraph, S_i , ($i=1...n$) that can be connected to the constraint graph, G , to which the constraint switch belongs.

PROPERTIES

1. A switch position is **selected** if it is connected to G , or if it has been specified to be connected to G .
2. Only one switch position can be selected at the same time.
3. The common constraint subgraph is connected to G whenever any switch position is selected, unless a null position is selected.
4. A constraint switch may or may not be required to have a switch position selected (i.e., a switch position that is equal to a null constraint graph generally is allowed).
5. A default switch position may be defined for a constraint switch which will be selected if no other switch position is selected.

6. Each constraint in a switch position constraint subgraph that is directly connected to G if the switch position is selected is known as a **pole**.
7. Each switch position may have a different number of poles (i.e., unlike most physical switches, a constraint switch can have a non-constant number of poles).
8. A common variable need not be constrained by each possible switch position.

DEFINITION 5.12 A **constraint schematic partition** is a constraint subgraph that is a constraint schematic.

The main purpose of a constraint schematic partition is to identify constraints and variables that form an option (defined next) in a constraint schematic. These constraints and variables should be connected to or disconnected from a constraint schematic as a group.

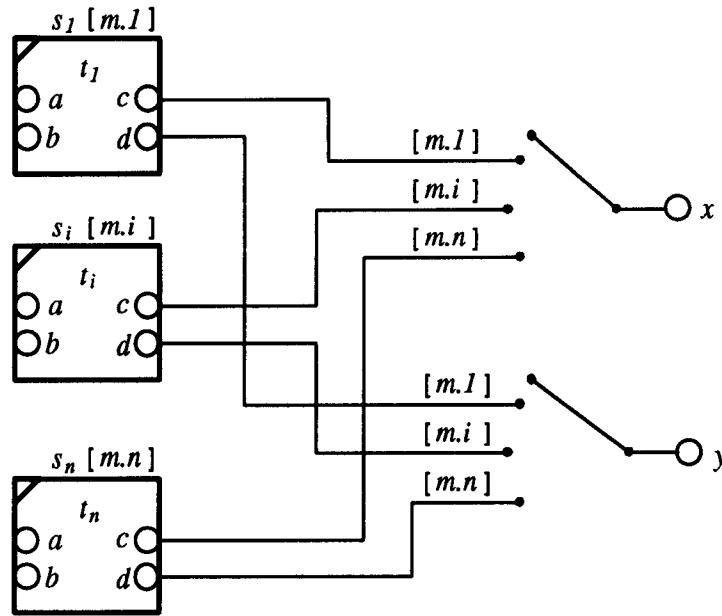
DEFINITION 5.13 An **option** is a switch position that is a constraint schematic partition.

DEFINITION 5.14 An **option category** is a constraint switch in which all switch positions are options.

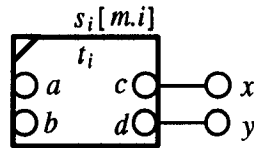
Specific option category examples are given later in the solder joint fatigue case studies (Chapter 9).

Subsystem substitution (Table 5.6) is a special type of ABB option category that allows alternate models of varying complexity level to be used (Objective 4 - Complexity Level). This subsystem substitution notation (Symbol S12) is equivalent shorthand for the literal constraint schematic shown in Figure 5.17. Option category m indicates subsystem s can be one of n possible types of objects, $s_i=t_i$, for $i=1...n$. Figure 5.17. Per the meaning of

option category m (switch m) in the literal constraint schematic, only one subsystem can be used at any one time. Each subsystem substitute, s_i , is an option (switch position) in the subsystem substitution option category (switch).



a. Literal Constraint Schematic



b. Specialized Notation

Figure 5.17 Subsystem Substitution Notation

The subsystem substitutes, s_i , typically represent alternate analysis models of varying complexity that share a common ancestor in the is-a hierarchy. For example, subsystem substitution could be used to offer greater accuracy in calculating transverse deformations in the same flat object. For this case possible subsystem substitutes, in order

of increasing complexity, are beams, plates, and shells. Their common ancestor in the is-a hierarchy would be thin deformable bodies.

As the ABB constraint schematic uses each possible subsystem substitute, s_i , for a similar purpose, the interface of each subsystem to the constraint schematic is nearly the same, if not exactly the same; thus, a high degree of modularity is possible. An actual example of subsystem substitution is given in the case studies (Chapter 9) where different PBAMs are used to provide strain input to a solder fatigue model.

5.7 EXTENDED CONSTRAINT GRAPHS

An extended constraint graph, introduced in Section 5.1, is another view of a analytical building block. It discards distinctions between types of relations and shows the detailed relations inside subsystems. Such characteristics can cause the "tangled knots" problem as noted previously, but this view can also be useful to trace constraint dependencies. Hence, one can use it visually to determine what variables are needed to produce a given output - something that is not immediately obvious given a constraint schematic with subsystems. Figure 5.18 shows the extended constraint graph of Elementary Rod. In the next example, this ABB is used as a subsystem with the view given in Figure 5.14 (b) (with $F=0$). Under this special case, the variables E and A are not needed as Eqn. 5.2 reduces to the following form:

$$\epsilon = \alpha \Delta T \quad (5.2')$$

Therefore, an extended constraint graph under this special case is given in Figure 5.19 where r_I' represents the preceding equation.

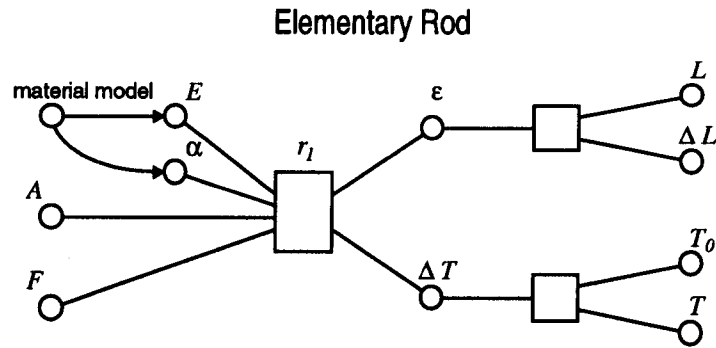


Figure 5.18 Extended Constraint Graph for Elementary Rod

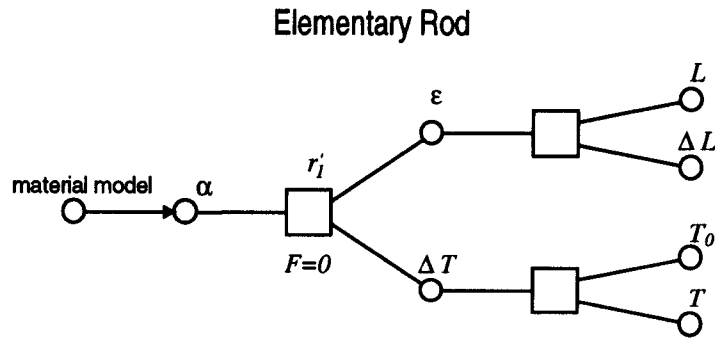


Figure 5.19 Special Case Extended Constraint Graph

EXAMPLE 5.3 ABB with a Subsystem (Safe Rod)

As a simplified example, assume an engineer is interested in the total deformation, ΔL_n , of the rod in Example 5.1 scaled by a safety factor, n . The engineer is only concerned about deformations caused by thermal loads. Figure 5.20 gives the constraint schematic of a new analytical building block, **Safe Rod**, for this purpose which uses **Elementary Rod** as a subsystem per the view given in Figure 5.14 (b) (with $F=0$).

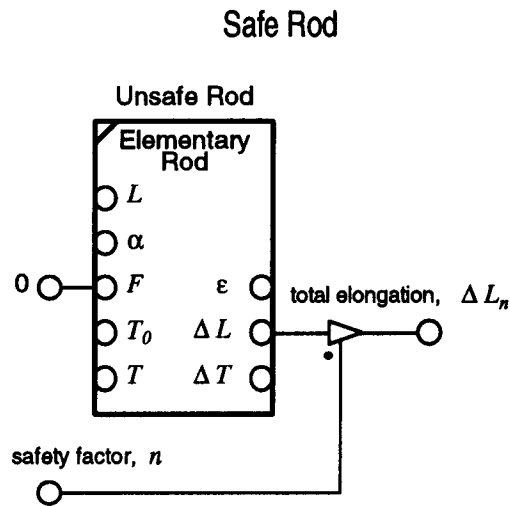


Figure 5.20 Constraint Schematic for Safe Rod

The extended constraint graph for Safe Rod is shown in Figure 5.21 along with an optional border around the portion that comes from the extended constraint graph of Elementary Rod (Figure 5.18). By comparing these last two figures, one can see that the constraint schematic is much simpler as it hides all the relations inside the Unsafe Rod subsystem. Though the extended constraint graph is more complicated, it enables one to trace through the dependencies among the variables.

For example, if one wanted to know how to obtain strain, ϵ , one can simply look for relations in which ϵ is involved (indicated by an edge between ϵ and a relation). Supplying inputs to the other variables in such a relation will then determine ϵ . For example, inputting ΔL_n and n will determine ΔL . Assuming one also inputs L , then ϵ will then be determined. Alternately, one could supply all inputs to r_1 except ϵ . Thus, there are several possible input combinations that will determine the value of variable ϵ .

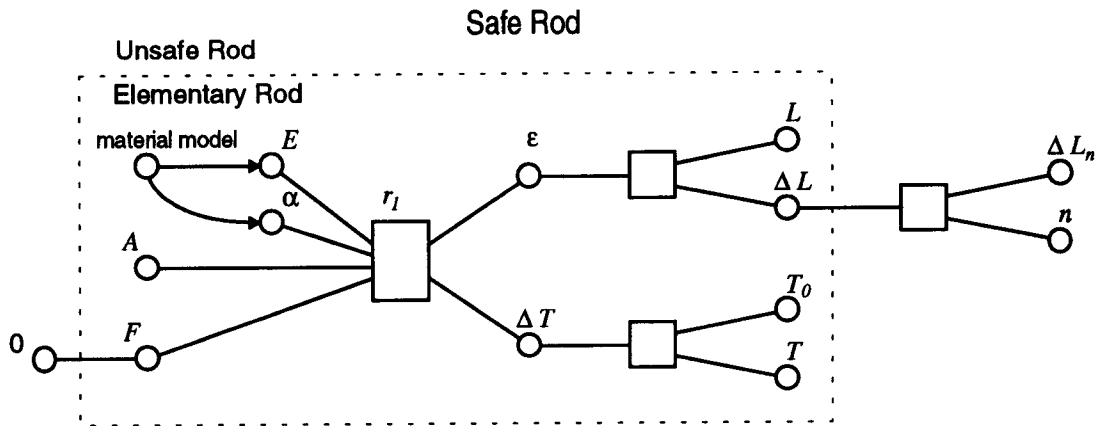


Figure 5.21 Extended Constraint Graph for Safe Rod

Finally, with complicated ABBs, it is probably best to show only one combination of options on a single constraint graph. Thus, *while an ABB has only one constraint schematic, it can have many extended constraint graphs* that correspond to potentially all combinations of options in each option category as well as for special cases (e.g., when a variable equals zero as in the above example).

5.8 I/O TABLES

Considering the example just covered, it would be convenient to have some way of documenting which inputs need to be supplied to give certain outputs. An **input/output table (I/O table)** is another view of an analytical building block that serves this purpose. This view is derivable from the ABB structure, but is obtained more easily by tracing the corresponding extended constraint graph. Ease of reference (particularly for the human user) is the primary utility of this view.

The general form of an I/O table is given in Table 5.7, where the attributes are explained in detail next. Briefly, the column headings, v_i , are the variables, and the rows

object name
Conditions
conditions

v_1	v_2	...	v_n
t	t	...	t
...			

Table 5.7 General Form of an I/O Table

are the input/output combinations (where $t = \text{I, O, I', m, a, or } \langle \text{blank} \rangle$ as explained below).

object The object to which the I/O table belongs.

conditions Any special conditions that are assumed (e.g., taking the value of a variable to be zero).

options The collection of options that are specified for this table.

variables An ordered collection of the variables under consideration. All variables must be attributes or subattributes of the specified *object*. These variables are the column headings in an I/O table.

I/O combinations A collection containing fully constrained I/O combinations (but not necessarily all possible combinations). An I/O combination is an ordered collection that is ordered according to the variables collection. In other words, each row in an I/O table is an I/O combination. Each element in the I/O combination indicates the I/O status of the corresponding variable according to the following symbols:

- I** The variable is an independent input. If the conditions include that a given variable must have a specific value, that variable falls under this category.
- O** The variable is the designated output. Only one variable can be the designated output in an I/O combination. Note that several I/O combinations can give the same variable as the designated output, O.
- I'** the variable is an independent input where one of its subvariables (in the part-of relationship) is the desired output, O.
- m** The variable is determined intermediately. In other words, the value of this variable is found in order to determine the designated output, O.
- a** The variable is determined by an ancillary relation (a relation that is not used to determine O but is still connected to the constraint graph and is fully constrained by the I/O combination).

<blank> The variable is not fully constrained by the given inputs in this I/O combination (i.e., its value cannot be determined from the given inputs).

Table 5.8 is a partial I/O table for Elementary Rod with respect to the subsystem view used in the Safe Rod example. The special case extended constraint graph in Figure 5.19 is useful for manually tracing the I/O combinations in this table. Note that several I/O combinations are shown which produce strain, ϵ , as an output.

Table 5.8 General Form of an I/O Table
Elementary Rod
Conditions
 $F=0$

L	α	T_o	T	ϵ	ΔT	ΔL
I	I	I	I	m	m	O
		I	I		O	
	I			O	I	
	I	I	I	O	m	
I				O		I
		I	O		I	
		O	I		I	
	O	I	I	I	m	
O				I		I

5.9 The ABB Structure

The ABB representation is an information model that has a defined *structure* and defined *operations* analogous to mathematical entities. For example, a matrix is composed of elements arranged in a row-column structure. Defined matrix operations include addition, multiplication, and inversion. This section defines the structure of the ABB representation using concepts that have been defined throughout this chapter. This structure is formally called the **ABB structure**. ABB operations are defined later in this chapter.

Several views of an analysis model have been defined thus far in this chapter. The ABB structure contains the same information as all these views combined. Each view discussed above is actually just a subset of the information model of an analysis model. The ABB structure contains the complete structural aspects of this information model. Therefore, one can say that the ABB structure is the *master view*, as all other structural views can be derived from it.

The general ABB structure can be thought of as a template with defined blanks to be filled in. Creating an ABB to represent a particular analysis model involves filling in those blanks with the relations and variables of that analysis model. The first part of this section defines these blanks.

DEFINITION 5.15 A relation is an **ABB-specific relation**, *ASR*, with respect to an ABB, *A*, if it is defined in *A*.

The general form of such a relation 1) is not generally useful or applicable outside the scope of *A* and/or 2) is not naturally part of another object (which is why *A* must define it). An example of case 2 is the linear elastic stress-strain relation for materials, which is an ABB-specific relation for a linear elastic material model ABB. Other ABBs can use this ABB (e.g., a Rod), but typically would not use the stress-strain relation apart

the context of the material model ABB. Therefore, as a counter example, the stress-strain relation would not be an ABB-specific relation with respect to a Rod. Counter examples for case 1 include the scale & offset relation and the absolute value relation which can be used in many different objects.

DEFINITION 5.16 A **linkage** is a relation in an ABB that is not an ABB-specific relation.

DEFINITION 5.17 A **semantic linkage** is a binary linkage that maps the name of a variable in one scope to a name that is meaningful in another scope. (This concept was first introduced in Section 5.5 with respect to subsystems.)

DEFINITION 5.18 An **ABB partition**, AP , is a constraint schematic partition that categorizes variables and constraints according to their use in the ABB structure. These subsets are:

variables, V	ABB-specific relations, ASR
subsystems, SS	linkages, L

One of main purposes of an ABB partition is to serve as a primary partition or as options in an ABB (see next definition).

DEFINITION 5.19 An **ABB (analytical building block)** is a representation of an analysis model. An ABB consists of a primary partition and a collection of option categories. In an ABB, the primary partition and all options contained in the option categories are ABB partitions.

OBSERVATION

1. The primary ABB partition is always connected to the constraint graph of the ABB.

The following abbreviated template summarizes the general structure of an ABB (Figure 5.22), where indentation denotes the part-of relationship. Also, $S=\{q_i\}$ denotes a set $\{q_1, q_2, \dots, q_n\}$ where all elements of S are of type q . The subscript i denotes a non-zero positive integer.

The general format used to document the ABB structure is given in Figure 5.23 where italicized names are items to be replaced with information that is specific to a given analysis model. In this template, an ellipsis (...) denotes possible repetition of the preceding entity. Figures 5.24 and 5.25 are example ABB structures that have been filled in with information specific to the Example 5.1 analysis model using the Elementary Rod constraint schematic (Figure 5.12) and associated object relationship diagram (Figure

ABB

```

primary partition, PP
  variables,  $V = \{\text{variable}_i\}$ 
  subsystems,  $SS = \{\text{abb}_i\}$ 
  ABB-specific relations,  $PSR = \{\text{constraint}_i\}$ 
  linkages,  $L = \{\text{constraint}_i\}$ 
  option categories,  $OC = \{$ 
    ABB option category $_i$ 
    common partition $_i$ 
    ABB options $_i$ ,  $OP_i = \{$ 
      ABB partition $_j$ 
        variables,  $V = \{\text{variable}_i\}$ 
        subsystems,  $SS_j = \{\text{analytical model}_k\}$ 
        ABB-specific relations,  $ASR_j = \{\text{constraint}_k\}$ 
        linkages,  $L_j = \{\text{constraint}_k\}$ 
      }
    }
  }

```

Figure 5.22 Abbreviated Structure of an Analytical Building Block (ABB)

5.11). A new type of ABB is designated as an ancestor class of the general ABB class (e.g., see Appendix F) by specifying its parent ABB class in the "Superclass:" slot.

Each *variable* and may have arbitrarily deep part-of hierarchies which can be specified in indented form (explained previously in this chapter for the part-of symbols G4 and S4). Each *variable* is a variable of the following form:

$$\text{variable: class (default: default value} \mid \text{:}=\text{ required value)}$$

The first designation, *class*, is a restriction on the type of variable that *variable* is, and the last two are restrictions on its value. All such designations are optional and based on EXPRESS syntax [ISO 10303-11]. Here the notation (*a* | *b*) indicates either *a* or *b* is allowed, but not both. A *subsystem variable* is a variable in a subsystem which follows the same syntax as *variable* and can have the same part-of hierarchies.

For subsystem variables, only those restrictions that are different or in addition to restrictions already specified in the subsystem models should be specified here. Note that these type and value restrictions are actually special types of unary constraints. Only subattributes used by the partition being described should be included.

The **Semantic Linkages** contains linkages of the type so indicated. Other linkages can be specified in the **Other Linkages** section by giving the relation itself using *variable* or *subsystem variable* names. Standard dot-delimited notation can be used to specify subvariables in these sections.

Option categories are defined in the so-designated section of the ABB structure. The notation [*q.k*] indicates that the variable or relation on that line belongs to option *k* in option category *q*. Items in the option category section that have no such designation belong to the common partition of that option category. Example usage of options is given in Chapter 9 and Appendix G.

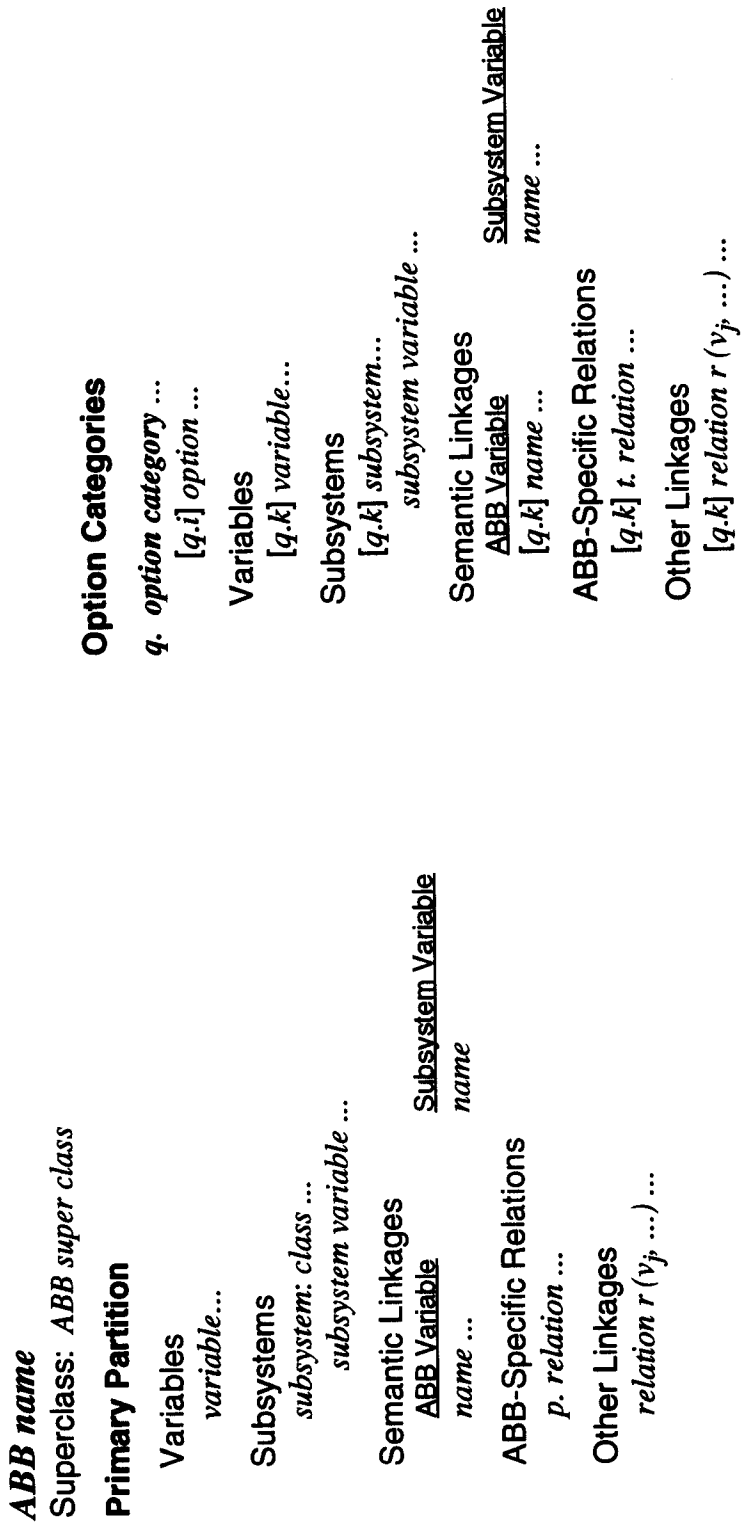


Figure 5.23 General ABB Structure

Comparison of Other Views

A constraint schematic graphically captures the class-oriented information that the object relationship diagram does not, including inherited *class values* of some class attributes (indicated by shading). However, the constraint schematic view is weak where object relationship diagram is strong: no is-a hierarchy information (beyond the shaded inherited attribute values) is depicted. The constraint schematic also does not show the restrictions on variable values contained in the ABB structure (class, default, and value restrictions). Although it identifies the existence of relations and how variables are connected to them, the constraint schematic does not give the definitions of the relations; the ABB structure supports such definitions. Altogether, with the exception of the items noted, the constraint schematic contains a large portion of the information defined in the ABB structure.

In summary, the ABB representation is a general information model of analysis models. This section has defined the general ABB structure. Creating a particular type of ABB that represents a specific analysis model involves filling in the ABB template with the specific information of that analysis model. The ABB structure can be thought of as the master view from which all other structural views can be derived.

5.10 SUMMARY OF STRUCTURAL VIEWS

Thus far, this chapter has introduced the structural aspects of the ABB representation as captured in the following five views:

Table 5.9 ABB Structural Views

- | | |
|-------------------------------|------------------------------|
| • ABB Structure | • Subsystems |
| • Constraint Schematic | • Extended Constraint Graphs |
| • Object Relationship Diagram | • I/O Tables |

The ABB structure fully defines a particular type of ABB that represents a specific analysis model. The ABB structure can be considered the master view in that all other structural views can be derived from it. As the ABB structure is in textual form, it can be difficult to comprehend as a whole. The other views aid comprehension through graphics visualization and information hiding, but they are incomplete. Therefore, all views together play complementary roles to help humans understand the structure of an ABB.

Appendix G shows one way in which all views can be combined together to provide a humanly comprehensible (but redundant) documentation of specific types of ABBs. This conglomeration could be shown without redundancies (e.g., the linkages in the master view could be omitted as they are shown in the constraint schematic) and is analogous to the "data sheet" description of electronic components in vendor data books.

For increased modularity and decreased redundancy, these ABB views reference the following kinds of resources:

1. Parent ABB "Data Sheets"
2. Other ABB "Data Sheets"

Therefore, to understand what a given ABB does, one may have to refer to such data sheets of other ABBs that are used to build the ABB of interest.

The importance of these views is that they can help one develop, implement, and use an ABB. *Developing* ABBs to represent a given analysis model involves populating the structural views given above and is largely implementation independent. Once the views are completed, the ABBs can be *implemented* in a computer system with the aide of mappings from the general ABB structure to a particular type of implementation. These views can help one understand how an specific ABB works and, hence, help one use an ABB. For example, an I/O table tells one what I/O alternatives are possible. The next

section deal with how one can use an ABB. Guidelines for both developing and implementing PBAMs are given in Chapters 7 and 8 respectively.


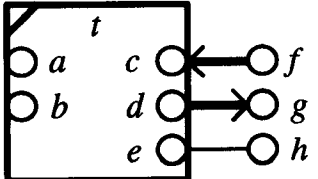
5.11 INSTANCE VIEWS

An **instance view** shows an ABB as *used* in a specific situation. It should be emphasized that, as the name implies, this view illustrates the potential use (*operation*) of an *instance* of a particular class of ABB; in contrast, the views defined previously deal with the general *structure* of a particular class of ABB. Table 5.10 gives the constraint schematic notation used to depict instance views.

DEFINITION 5.20 An **analysis context** is a user of an ABB instance. This user can be either a person or a computer tool (e.g., an expert system that needs some analysis results). Another ABB cannot be an analysis context.

To utilize an ABB once it has been implemented, the analysis context initializes the ABB, connects the necessary analysis entities, and specifies which variables in the ABB are inputs, as well as which variable is the output. (These operations that an analysis context can perform on the ABB instance are defined in the next section). If the analysis

Table 5.10 Constraint Schematic Instance View Notation

S13	<p style="text-align: center;">Jumper</p> 	The analysis context has connected <i>a</i> and <i>b</i> together using a jumper .	$a \neq b$
S14	<p style="text-align: center;">Instance View</p> 	An <i>instance</i> of subsystem <i>s</i> has variable <i>f</i> input into variable <i>s.c</i> . Variable <i>g</i> is read as an output from <i>s.d</i> . In contrast, variables <i>h</i> and <i>s.e</i> are always equal.	

context is a person, highly interactive use of the ABB may be possible. For example, the person can change which variables are inputs and which are outputs to answer "what-if" type questions. Additionally, he or she can inject new values into the same input/output combination and see the result by probing other variables.

DEFINITION 5.21 A **jumper** is an equality relation that is not part of the defined constraint schematic of an ABB. Only an analysis context can add and delete jumpers.

Constraint schematic jumpers are analogous to physical jumper wire connections used to test electronic systems (thus the name of this concept and Symbol 13). They can be added and deleted dynamically by an analysis context, whereas the constraint schematic of a specific class of ABB is static, barring any correction or enhancement to its ABB structure. Jumpers can be thought of as non-permanent connections; however, they are not removed until the analysis context removes them. Two types of instance views that use jumpers are described next.

Standard Instance View

A **standard instance view** contains a subsystem view of an instance of ABB along with its connections to an analysis context. Using the standard instance view notation (S14), a representative form of this view is given in Figure 5.26, for ABB_0 - an ABB of type t . If desired, specific values of variables can be included, as well as arrows indicating the input/output combination chosen.

The analysis context specifies the options for ABB_0 , which can be indicated in a (possibly) indented list. If the ABB contains nested subsystems with option categories, the instance view must specify those options as well, or default options will be used.

Thus, this option list will generally be hierarchical, where indentation of a given line indicates how deeply nested the subsystem is for which the option is specified.

For example, in Figure 5.26 the options indicated for ABB_0 are p.2 and q.1 (where p and q are the names of two option categories in ABB_0 , and the numbers 2 and 1 indicate the selected option within each category). Option category p might be a subsystem substitution type category. In the figure, the analysis context has selected option p.2 to specify a specific ABB (designated as option #2) to use as the substituted subsystem. This subsystem evidently has an option category r, and option r.1 has been selected.

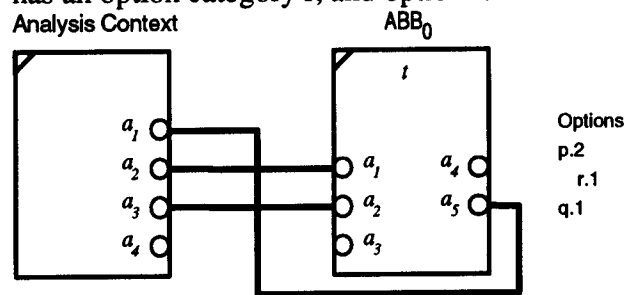


Figure 5.26 Representative Standard Instance View

Detailed Instance View

In a **detailed instance view**, the ABB subsystem symbol graphically includes the constraint schematic of the ABB to the extent that is needed for the analysis context at hand. In other words, this view is basically a standard instance view with some of the inner ABB details shown.

Switch positions and the names of selected substitute subsystems can be changed to indicate which options are in effect for a detailed instance view (i.e., the switch positions and subsystem names are changed from the neutral positions and names that they have in the ABB constraint schematic). Values of intermediate variables within the constraint schematic can also be shown on this view if desired.

Chapter 9 contains examples from the case studies of both standard and detailed instance views.

5.12 Preliminary ABB Operations

Whereas the previous sections defined the *structure* of the ABB representation, this section discusses the *operations* of the ABB representation. The general process for an analysis context to use an ABB is given in Figure 5.28, which is an adapted version of the general routine analysis process given in Chapter 4 (Figure 4.4). This figure shows the steps that the analysis context performs to use a particular type of ABB, e.g., as in instance views defined in the previous section. Each step designated by a dagger (†) in the figure can be carried out by one or more operations that are part of the ABB representation. Those steps with no symbol are beyond the scope of the current ABB representation.

At present, the general definitions of the operations that fulfill each designated step

1. Design product.
 2. Perform routine analysis.
 - 2.1 Identify application (the design problem).
 - 2.2 Choose ABB for application.
 - † 2.3 Setup ABB.
 - † 2.3.1 Instantiate ABB.
 - † 2.3.2 Specify options.
 - † 2.3.3 Link ABB with input/output entities.
 - † 2.3.4 Specify I/O combination.
 - † 2.4 Reconcile ABB.
 - † 2.5 Read results.
 - 2.6 Check results.
 - 2.7 Interpret results to determine design changes.
 3. Make design changes.
- † Analysis context responsibilities (performed using ABB operations).

Figure 5.27 Steps for Using ABBs in Routine Analysis

have not been developed. Therefore, the following discussion describes these operations in general terms based on preliminary work done to date.

Steps for Initial Use of an ABB

STEP 2.3 Setup ABB.

To fulfill this step, the analysis context performs the substeps given in the figure (Steps 2.3.1-2.3.4), which are defined below.

STEP 2.3.1 Instantiate ABB.

This step creates an instance of the specified type of ABB (i.e., a template of the ABB is created per the ABB structure of this type of ABB - at this point it does not contain specified options and the *values* of the variables are not known). Upon instantiation, operations internal to the ABB recursively initialize the primary partition by initializing all primary partitions in the ancestor classes of the ABB.

Initializing an ABB partition is a basic operation also used in other steps that involves the following substeps:

1. Instantiate each subsystem in the partition (i.e., recursively performing this operation (Step 2.3.1) on each subsystem in the partition). Note that this process continues with each nested subsystem as deep as is needed (i.e., until subsystems with no internal subsystems are reached).
2. Create constraints between subsystems to fulfill the ABB-specific Relations and the Other Relations specified in the ABB structure.

Thus, recursion is used both to initialize the primary partition in an ABB (recursively up the is-a hierarchy) and to instantiate each subsystem in the partition (recursively down the subsystem nestings).

STEP 2.3.2 **Specify** options.

The analysis context specifies options by first designating the options for the outermost ABB (i.e., the ABB that was chosen in Step 2.2). Alternately, if the analysis context does not specify any options, default options will be used.

Internally, the ABB initializes the common partition of each option category and of each specified option. Then, depending on the types of options in the ABB, the analysis context may designate options for ABBs that serve as subsystems in the newly initialized partitions. These nested option specifications can go as deep as the subsystem nestings, and tend to occur with option categories that are of the subsystem substitution type.

STEP 2.3.3 **Link** ABB with input/output entities.

The analysis context performs this step by assigning an entity to each desired variable in the ABB (i.e., the entity becomes the *value* of the associated variable). Note that entities can be assigned to variables that will be the output and/or intermediate results in the ABB for the purpose that their values can be received as outputs. Thus the term "link" is used above rather than just saying "*Input* the needed (input) entities into the ABB."

STEP 2.3.4 **Specify** I/O combination.

The analysis context tells the ABB which variables are the inputs and which is the output in order to designate which variables in the ABB *are not* allowed to change (the inputs, **I**) and which *are* allowed to change (the output, **O**, and all other variables not specified as inputs). The I/O table corresponding to the selected options is helpful to determine which I/O combinations are possible for this step. In fact, potentially the ABB can use such tables to check that the specified I/O combination is valid.

STEP 2.4 **Reconcile ABB.**

In this step the analysis context basically tells the ABB that its setup has been completed and, hence, that the ABB should determine the output based on the given inputs and designated options. In other words, the ABB is told to reconcile the values of all variables by making them consistent with the given inputs and designated options.

STEP 2.5 **Read results.**

The desired result can be seen by retrieving the value of the output variable. Similarly, intermediate and ancillary results (designated by **m** or **a**, respectively, in the I/O table) can be seen by retrieving the values of their associated variables.

Steps for Subsequent Interaction with an ABB

After Step 2.5 has been completed, the analysis context can proceed to the subsequent steps as given in Figure 5.28, or it may perform some of the following steps to interact with the ABB. These steps and associated operations are useful for "what if" type design scenarios as exemplified in Chapter 9. The prime (') next to the step number indicates that essentially the same step as above is being performed, but that the ABB has been reconciled (Step 2.4) at least once. Therefore, the below steps are ones that *change* an existing ABB instance rather than setup a new ABB instance for the first time. Consequently, Step 2.3.1' does not exist.

STEP 2.3.2' Change options.

It may be possible to change options in the current ABB instance, for example, to see the effects of different loading conditions. The internal workings to achieve this step (*changing* options) have not been investigated in this thesis.

STEP 2.3.3' Change ABB linkages with input/output entities.

This step can be performed to try new input values in the previously specified I/O combination, or to create new links with new inputs/outputs for a new I/O combination.

STEP 2.3.4' Change I/O combination.

After the analysis context specifies a new I/O combination (i.e., *changes* the I/O combination) in the same manner as before, internally the ABB must change inputs and outputs in such a way that the state of a given variable is not changed unless so desired. In other words, if a variable that is currently an output is changed to be an input, the value of the variable that was determined should be preserved, if desired, so that the same value can be used subsequently as an input.

STEP 2.4' Reconcile ABB.

This step is basically the same as before.

STEP 2.5' Read results.

This step is basically the same as before.

5.13 Preliminary Set of General Purpose ABBs (GPABBs)

The section highlights the initial set of **general purpose ABBs (GPABBs)** (i.e., ABBs that are not inherently associated with a particular type of product) that are described in detail in Appendix F. All ABBs introduced thus far are classified as general purpose ABBs. Table 5.11 summarizes the different types of GPABBs discussed in Appendix F³, which also includes ABB structural views of for some of these GPABBs.

Table 5.11 Categories of General Purpose ABBs (GPABBs)

<i>GPABB Category</i>	<i>Examples</i>
• Matter Models	• Linear Elastic, Viscoplastic, Coffin-Manson
• Continuum Models	• Shell, Plate, Beam, Rod
• Discrete Elements	• Spring, Mass, Resistor
• Interconnection Models	• No-Slip Connection, Revolute Pair
• Analytical Variables	• Force, Stress, Strain, Temperature
• Analytical Systems	• Cantilever Beam System

It should be emphasized that this initial set is not the focus of this research; rather *the intent has been to define such GPABBs to the extent that they can be used by PBAMs*, per the methods given in Chapter 6. (The reader is encouraged to skim Appendix F now for a better appreciation of the PBAM discussion in the next chapter). Figure 5.28 is an object relationship diagram containing some of these GPABB analytical primitives (**Beam** and **Linear Elastic Model**) as used to create exemplar analytical systems (**Slender Body Systems**).

³ The reader is encouraged to skim Appendix F at this point to enable a better understanding of the material to come in Chapter 6.

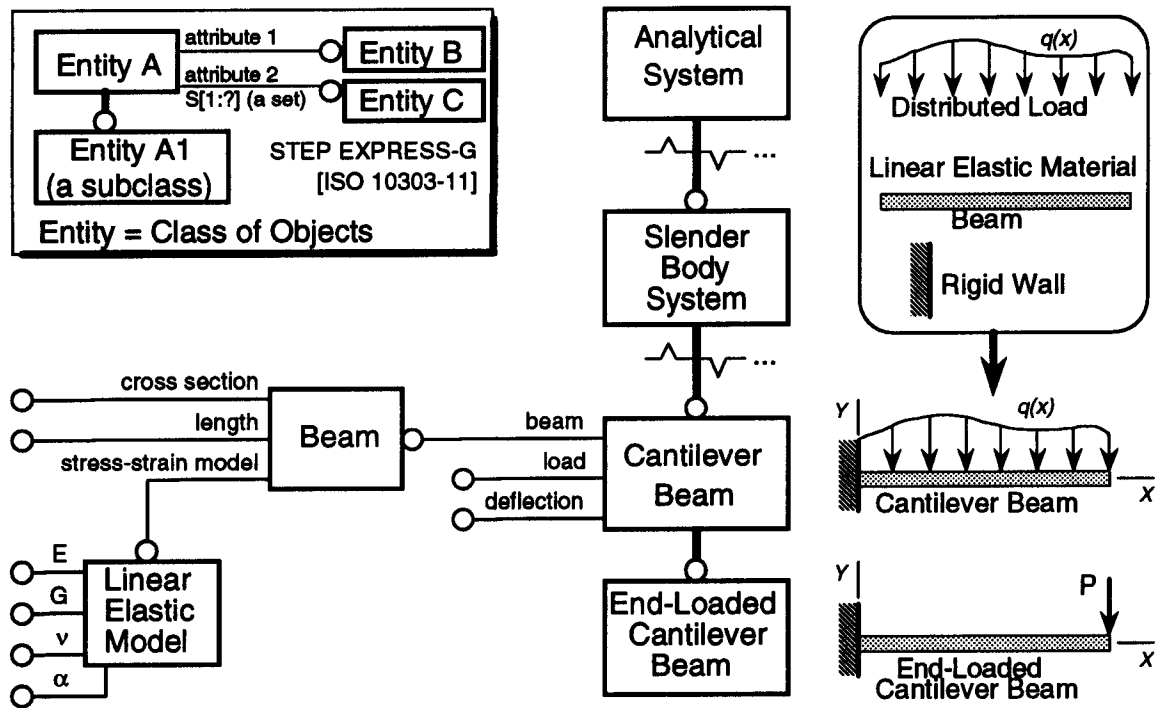


Figure 5.28 Example GPABB Object Relationship Diagram

EXAMPLE 5.4 Matter Model (HIH Model, a.k.a. Linear Elastic Model)

Figure 5.29 shows the partial constraint schematic and one possible subsystem view for the isotropic linear elastic (HIH) stress-strain model HIH Model (a type of Matter Model), used in Figure 5.28. Relations among the material properties as well as part of the constitutive relation between stress and strain are shown [Crandall, et al., 1978].

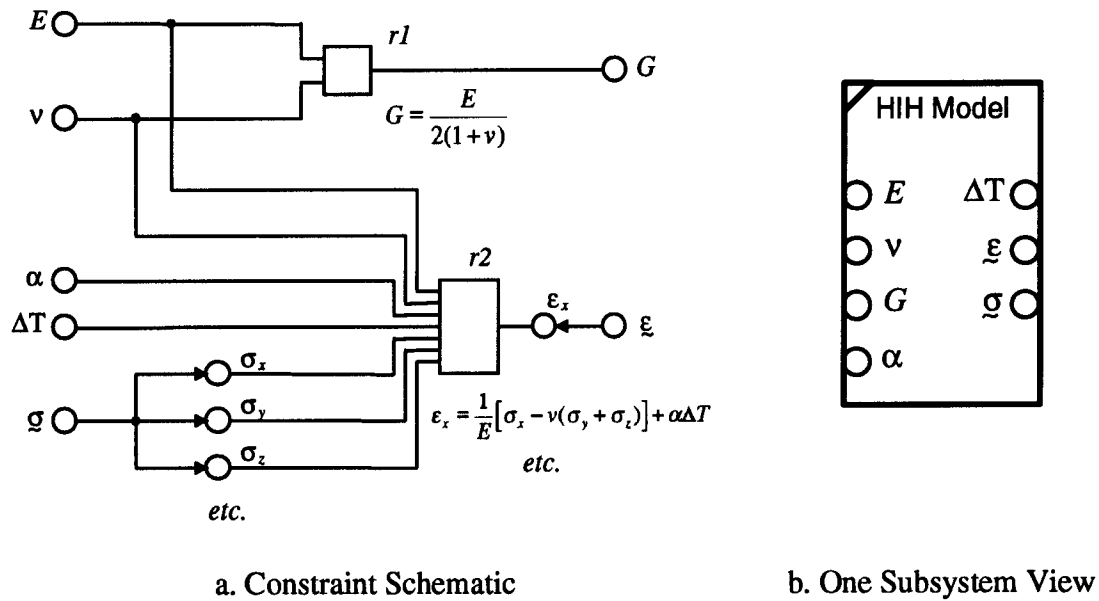


Figure 5.29 A Matter Model

5.14 Summary

This chapter defined the structure and operations of the analytical building block (ABB) representation to represent engineering analysis models. Several views of this structure were introduced using simplified examples, including the constraint schematic view which merges constraint and object concepts. The master view, termed the ABB structure, contains the complete structural definition of an ABB; all other structural views can be derived from it.

Operations supported by the ABB representation were given that define how a user can interact with an ABB. One operational view, the instance view, was provided which depicts such ABB usage. Finally, an overview was given of the starter set of general purpose ABBs contained in Appendix G.

CHAPTER 6

THE PBAM REPRESENTATION

The previous chapter defined the ABB representation which can be used to represent engineering analysis models. Two mutually exclusive types of ABBs can be defined:

1. Non-product-specific ABBs (i.e., General Purpose ABBs)
2. Product-specific ABBs (i.e., PBAMs)

The ABBs in the first category are "general purpose" in the sense that they potentially can be used in the analysis of many different types of products. In other words, by themselves general purpose ABBs contain no linkages with a specific type of product. (All examples given in the preceding chapter are general purpose ABBs.)

In contrast, product-specific ABBs contain such associativity by definition; such ABBs are called **product model-based analytical models (PBAMs)**. The term analytical building block (ABB) has been used in generic reference to both types of ABBs because PBAMs generally are built from both types of ABBs. Figure 6.1 summarizes the different categories of ABBs using indentation to indicate the is-a relationship. The two terms in this list that have not been discussed yet (Simple PBAM and Complex PBAM) are defined later in this chapter.

```
graph TD
  ABB[ABB] --> G[General Purpose ABB]
  ABB --> AP[Analytical Primitive]
  AP --> AS[Analytical System]
  AP --> PBAM[PBAM]
  PBAM --> SPBAM[Simple PBAM]
  PBAM --> CPBAM[Complex PBAM]
```

Figure 6.1 Classification of Analytical Building Blocks

This chapter first overviews product modeling concepts and then discusses how product models can be linked with analysis models using PBAMs. The general PBAM representation is then defined in terms of how it is a specialization of the ABB representation. Because of this relationship, the same ABB structure, operations and associated views (Subsystems, Constraint Schematic, Extended Constraint Graphs, Object Relationship Diagram, I/O Tables, and Instance Views) that were described in the previous chapter apply to PBAMs as well. Special considerations when using these views for PBAMs are also given in this chapter.

6.1 Product Model Background

Since PBAMs deal directly with detailed design information, an overview must be given of the product modeling approach to represent such information. ISO STEP (the **ST**andard for the **E**xchange of **P**roduct **M**odel **D**ata) [ISO 10303-X] introduced in Chapter 4 is perhaps the most extensive work to date that deals with the representation of product life cycle information. The titles of representative Parts of STEP in Table 6.1 convey the wide variety of such information, including geometry, material, assembly information, cost, and versioning. (See the References for a complete listing of STEP Parts.)

STEP, based on a building block philosophy, defines Integrated Resources that are used to build product models for specific purposes as defined in the Application Protocols (APs) [ISO 10303-1]. The Application Resources in the Integrated Resources section are product models for a particular class of product information (e.g., Part 102 Ship Structures) which are built largely from entities defined in the Generic Resources. In a nutshell, STEP is an attempt to define a large library of entities from which product models can be built. Some product models for specific product domains are included in this library.

Table 6.1 Representative STEP Parts for Product Models

Integrated Resources

Generic Resources

10303-41,	Part 41: Fundamentals of Product Description and Support
10303-42,	Part 42: Geometric and Topological Representation
10303-43,	Part 43: Product Shape Interface Model
10303-44,	Part 44: Product Structure Configuration Management
10303-45,	Part 45: Materials
10303-46,	Part 46: Presentation
10303-47,	Part 47: Shape Tolerances
10303-48,	Part 48: Form Features
10303-49,	Part 49: Product Life Cycle Support

Application Resources

10303-101,	Part 101: Drafting
10303-102,	Part 102: Ship Structures
10303-103,	Part 103: Electrical Applications
10303-104,	Part 104: Finite Element Analysis
10303-105,	Part 105: Kinematics

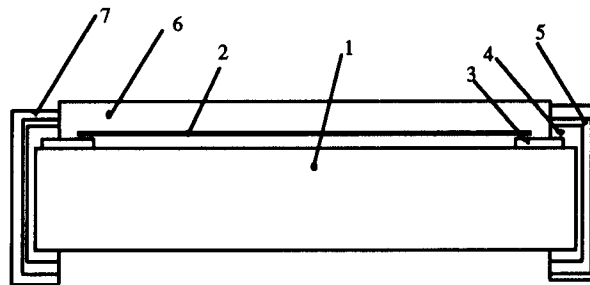
Application Protocols

10303-203,	Part 203: Exchange of Configuration Controlled Data
10303-207,	Part 207: Sheet Metal Dies Planning and Design
10303-208,	Part 208: Life Cycle Product Change Process
10303-209,	Part 209: Composite and Metallic Structures

A basic assumption taken in this research follows: *A product model must exist that represents the product being analyzed.* The product information needed by the analysis model is extracted from this product model - PBAMs specify what information is extracted and what part of the product model it comes from. As product modeling is still a relatively new field, it is more likely that a particular product model does *not* exist that contains all the information needed by a particular analysis model. Therefore, at present, the development of a particular PBAM to represent a particular analysis model, and the development or extension of a particular associated product model probably will require simultaneous effort. The STEP project is viewed as a potentially major source of modeling techniques and product entity resources for such product model development;

however, as many of the Parts listed in Table 6.1 are still in early development stages, the ultimate usefulness of STEP remains to be seen.

EXAMPLE 6.1 Product Model (Surface Mount Resistor)



1 - Ceramic base (96% Al_2O_3); 2 - Resistive element; 3 - Land termination;
4 - End cap (electrode metal); 5 - Nickel barrier; 6 - Protective glass film; 7 - Solderable coating

Figure 6.2 Construction of a Thick Film Rectangular Chip Resistor

From a PWA design perspective, an electrical component is just a simple device that has only a few attributes of interest (e.g., part number, cost, and total length width, height). In actuality, electrical components themselves are "assemblies" that can be relatively complex as illustrated by the construction of the surface mount resistor in Figure 6.2 [Mao and Fulton, 1992; Hinch, 1988, p. 34]. A simplified product model of this device is given in Figure 6.3 that will be used in later PBAM examples. Note that this figure is only an object relationship diagram view (using EXPRESS-G) of the product model.

The ceramic base is modeled as a Rectangular Primitive Component - a physical object composed of a single material upon which other Primitive Components can be added/assembled. The material attribute of Primitive Component has an attribute of type Elementary Material Model (a linear elastic stress-strain model) which is general purpose ABB defined in the last Chapter in Example 5.1.

All other aspects of the chip resistor are modeled as **Additive Layers** which is a special type of **Basic Component** that represents thin structures such as coatings and films. Whereas attributes describing the region covered these layers should be included, only the **thickness** is included in this product model for simplicity.

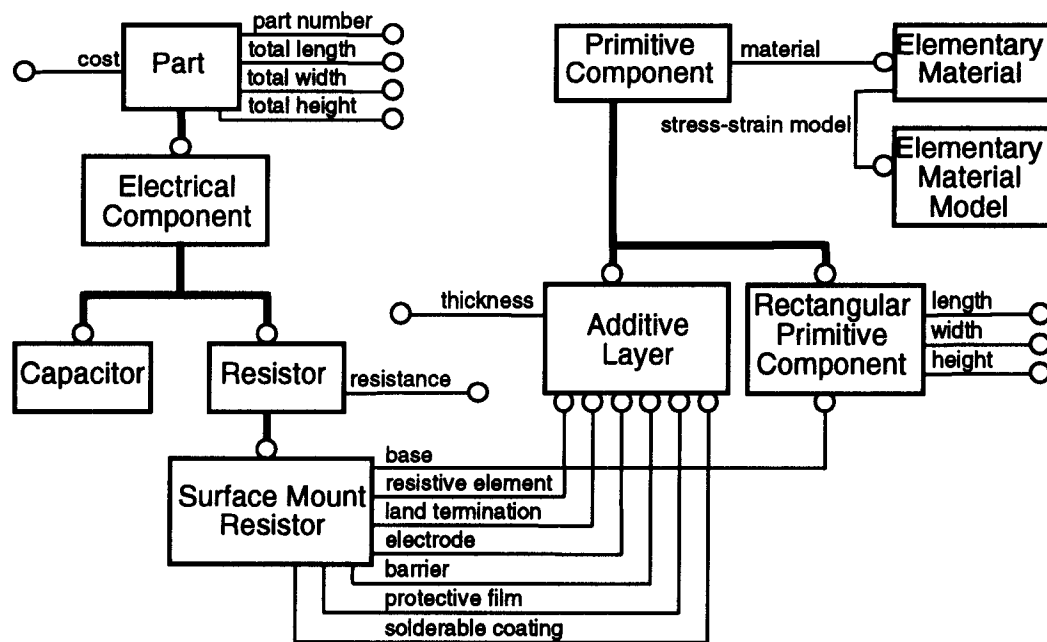


Figure 6.3 Simplified Product Model of a Surface Mount Resistor

A key point from this example is that even a relatively simple device like a chip resistor can have a rather detailed and complicated product model. Whereas an analysis model of a full PWA might need only a small part of this component information, a detailed analysis model of a solder joint connected to such a component might need much more.

A final key point is that product model attributes also can have relations among them. For example, the **total length** attribute of a **Part** can be determined by the following

relation, where L means length, t means thickness, and the subscripts refer to the region numbers in Figure 6.3:

$$L_{total} = L_1 + 2(t_4 + t_5 + t_7) \quad (6.1)$$

Such relations can be used to supply information to an analysis model as discussed in the next section.

6.2 Product Model Transformations for Analysis

Objective 3 (Associativity) in Chapter 4 covered the need to relate product information to analysis information along with the characteristics of this associativity. Analysis modeling idealizations and design synthesis operations were described there as transformations that, respectively, convert product information into analysis information and vice-versa.

One key concept to be established in this section is that modeling idealizations and design operations can be viewed as opposite forms of the same relation. If viewed as a binary relation, a modeling idealization is the standard form of the relation (i.e., the "natural" form - with respect to an analysis perspective) The corresponding design operation is then the inverse form of the same relation. Therefore, the term **product-analysis transformation (PAT)** is used here for a single relation that has both standard and inverse forms. For example, knowing the material in a part and determining the linear elastic stress-strain properties of that material is the standard form (an idealization) of a PAT. The inverse form (a design synthesis operation) of the same PAT follows: given some desired stress-strain model properties, find a material that has those properties.

As a relation, a PAT can be included in the constraint schematic of a PBAM just as any other relation. With this approach, *bi-directional associativity* between analysis information and product design information is possible. Before the term PAT is defined

specifically, a more precise distinction must be made between product and analysis information.

DEFINITION 6.1 A **product variable** is a variable that a designer would specify in order to define a product fully so it can be manufactured.

Admittedly, this definition is not that formal, but it serves its intended purpose in this chapter. A product variable usually cannot be used directly by an analysis model but instead may be an aggregation of variables that can.

For example, an entity representing a material such as solder can contain numerous models of its stress-strain behavior. The material entity is a product variable as a designer can specify a particular type of material. In contrast, specifying material properties in and of itself does not determine what material should be used to make a product. Each of the stress-strain models and the attributes they contain (e.g., E , ν , G , and α for an isotropic linear-elastic material model) are considered *analytical variables* (see the next DEFINITION). Analysis models would utilize entities that represent stress-strain models rather than those representing materials.

PROPERTY

1. A product variable in a PBAM is **patriarchal** if it is not a uniquely specified attribute of another product variable in the PBAM.

REMARK Given a collection, E , that is a product variable in a PBAM, P , and a product variable, e , such that $e \in E$. It is possible to have e as a patriarchal product variable in P for any $e \in E$.

Specifying E does not uniquely specify e . Therefore, e meets the criteria of PROPERTY 1 above. Therefore, the preceding REMARK is true. The purpose of

this property and remark is to establish that both a collection and its members can be "independent" product variables in the same PBAM. The only restriction on e is that it be a member of E .

An example of this situation occurs in the solder joint fatigue case studies. Briefly, a PBAM has a product variable called `pwa` that has a collection called `component occurrences`. The same PBAM also has a product variable called `component occurrence` whose value specifies the component-solder joint-pwb assembly to be analyzed. The analysis context must specify which component occurrence in the collection will be assigned to the variable `component occurrence`, as specifying a value for `pwa` does not specify which component occurrence to analyze.

DEFINITION 6.2 An **analytical variable** is a variable that is used by an analysis relation.

All variables in general purpose ABBs are analytical variables, while a PBAM can contain both types of variables. Examples of analytical variables include boundary conditions, performance variables, simplified geometry, and solution method parameters. Some analytical variables may be defined for convenience in linking subsystems and PBAM-specific relations (e.g., when a variable is used intermediately by more than two subsystems or PBAM-specific relations).

DEFINITION 6.3 A **product-analysis transformation (PAT)** is a linkage between one or more product variables and one or more analytical variables.

Some noteworthy aspects of PATs, such as their standard and inverse forms, were discussed at the beginning of this section. Another one is that PATs usually are contained in the product model if they are PATs that are commonly used. For example, a relation to

calculate total dimensions of a multi-material part or an assembly is a PAT that would be useful in potentially many PBAMs. Such a relation was encountered in Example 6.1 for the surface mount resistor (see Eqn. 6.1). As in that example, a part-of relationship will exist between the analytical variable (e.g., total length) and product variables in the product model. PATs that are defined in the product model are called **standard PATs**.

PATs that are defined in a particular type of PBAM are a special type of ABB-specific relation and are called **PBAM-specific PATs**. Like general ABB-specific relations, they are not generally useful outside the PBAM which defines them.

6.3 Simple PBAMs

The preceding section established that a PAT is a linkage which represents a modeling idealization and associated design synthesis operation as a single relation. Hence, a PAT relates product variables with analytical variables. This section shows how these newly defined concepts fit into the PBAM representation to make it a specialization of the ABB representation.

The basic structure of a simple PBAM is illustrated in the form of a detailed instance view in Figure 6.4. (The difference between a "simple" PBAM and a "complex" PBAM will be defined in the next section). Typical categories of each type of entity contained in a simple PBAM are included.

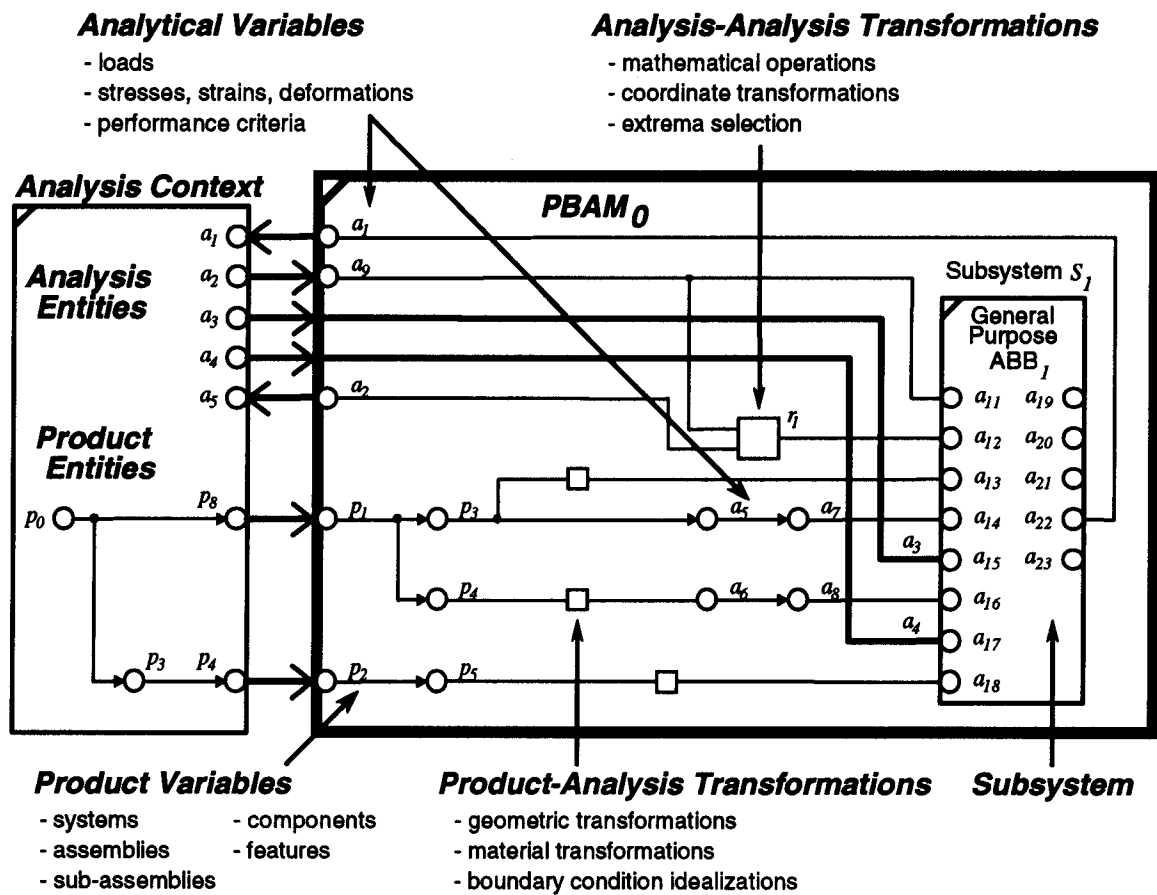


Figure 6.4 Structure of a Simple PBAM

A PBAM has one or more **patriarchal product variables** (p_1 and p_2 in PBAM₀ in this figure) which specify the eldest product variables that can be connected to the PBAM. Internally the PBAM can decompose these variables into their subattributes (p_3 , p_4 , p_5 in PBAM₀) per the part-of relationship. **Analytical variables** (a_1 through a_8 in PBAM₀) can also be included in the scope of the PBAM and may be decomposed into subattributes in the same manner as product variables.

Product-analysis transformations (PATs) are used to link product variables with some of these analytical variables. For example, a PBAM-specific PAT exists between p_4

and a_6 . A standard PAT is evidently contained the product model as a_5 is related to p_3 via the part-of relationship.

Because a PBAM is an ABB, it can contain general purpose ABBs as exemplified by the presence of subsystem s_I . Both product variables and analytical variables in the scope of the PBAM are typically eventually related to the analytical variables to such a subsystem. For example, a PBAM-specific PAT exists directly between p_5 and subsystem variable $s_I.a_8$.

Also, there may be **analysis-analysis transformations** within the PBAM scope that relate analytical variables (e.g., r_I in PBAM₀). This type of transformation has the following definition:

DEFINITION 6.4 An **analysis-analysis transformation (AAT)** is a linkage that relates two or more analytical variables.

The emphasis of this type of linkage is relating analytical variables within a PBAM scope and transforming analytical variables between subsystems.

EXAMPLE 6.2 Simple PBAM (Component Extensional Model)

This example assumes that one wished to determine the total deformation of an electrical component under a thermal load. Example 6.1 discussed how one such electronic component (a surface mount resistor) is a non-trivial, multi-material fabrication.

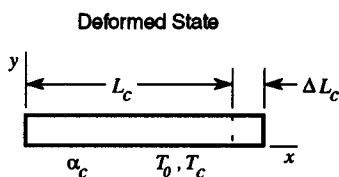


Figure 6.5 An Electrical Component Analysis Model

As a first approximation, one can treat such components as homogeneous rectangular body that undergoes uniform extension (Figure 6.5). The following equation then can be used to determine the total component deformation, ΔL_c :

$$\Delta L_c = \alpha_c (T_c - T_o) L_c \quad (6.2)$$

where L_c is the total length of the component which can be found using a relation like Eqn. 6.1. Here α_c is the CTE of the primary material in the component. In the case of a surface mount discrete (SMD) resistor like that in Example 6.1, the substrate can be assumed to be the major body that contributes to the thermally induced deformation (due to its geometric size and the magnitude of its material properties). Such an assumption is based on qualitative engineering judgment. Thus, for SMD resistors α_c equals the CTE of the substrate material modeled with an isotropic linear elastic stress-strain relation. This relation can be written as follows:

$$\alpha_c = \text{component.primary CTE material.}\alpha \quad (6.3)$$

Before constructing a PBAM to represent this analysis model, it is assumed that a product model (like that in Example 6.1) exists or can be developed for all components to be considered with this analysis model. As this simple component analysis model assumes extension, it is natural to investigate if the Elementary Rod ABB (Example 5.1) can be used to build a PBAM of this analysis model (because this general purpose ABB represents the common engineering concept "rod," which is a body with uniform extensional behavior).

By looking at the relations and variables in an Elementary Rod, one sees that Eqn. 6.2 is a special case of this ABB's relations in which $F=0$. Thus, one can use this general purpose ABB as a subsystem in a simple PBAM to represent the above component analysis model. Figure 6.6 is the resulting constraint schematic of this new PBAM which is called a Component Extensional Model.

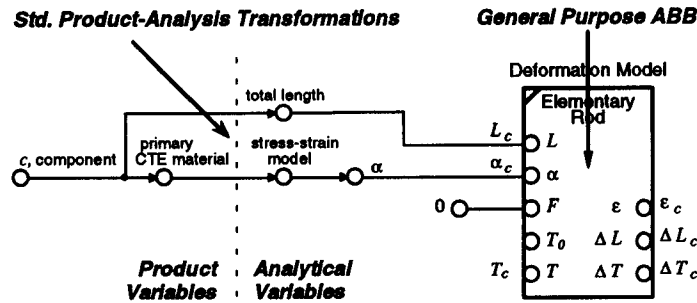


Figure 6.6 Constraint Schematic for Component Extensional Model

Here the subsystem is of type Elementary Rod and has been named Deformation Model per the engineering semantics of the component analysis model. The special case subsystem view where $F=0$ has been used (from Figure 5.14.b).

To link necessary product information with this subsystem, two standard PATs have been used. Note the division between product variables and analytical variables at these PATs as indicated by the dashed line. The top PAT deals with the total length attribute of component and utilizes the total dimensions PAT given as an example when PATs were defined in the previous section. This PAT is Eqn. 6.1 for surface mount resistors. The other PAT determines the primary CTE material of the component (e.g., using Eqn. 6.3) and extracts its stress-strain model. Finally, the needed connection between the CTE attribute of the stress-strain model and the subsystem can be made. This PAT and attribute was not originally included in the resistor product model, but they can be added readily.

One possible subsystem view of this PBAM is given in Figure 6.7 which meets the original intent of the analysis model (to determine component deformation). Because general purpose ABB was used, other analysis outputs are also possible from this PBAM, including component strain, ϵ_c , and component temperature change, ΔT_c , (as seen in the

constraint schematic). In summary, this example shows how a simple PBAM can be built from a general purpose ABB and illustrates some of the major aspects of a PBAM: product variables, analytical variables, PATs, and AATs.

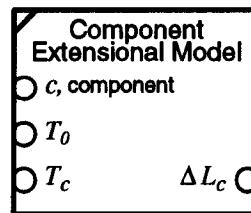


Figure 6.7 Example Subsystem View for Component Extensional Model

6.4 Complex PBAMs

Because PBAMs are ABBs, PBAMs can be used as subsystems in other PBAMs. This nesting of PBAMs can be arbitrarily deep. Thus, it is helpful to distinguish between **simple PBAMs** and **complex PBAMs** by the characteristics given in Table 6.2.

Table 6.2 Characteristics of Simple and Complex PBAMs

	# of Subsystems	Subsystem Types	Emphasis
Simple PBAM	1	General Purpose ABBs	Product-Analysis Transformations
Complex PBAM	1 or more	Any ABB (including PBAMs)	Subsystem Interactions

The structure of a complex PBAM is illustrated in Figure 6.8 which shows how a complex PBAM can contain multiple subsystems which may themselves be other PBAMs. General purpose ABBs may be used as subsystems and require analytical variable connections just as when used in a simple PBAM; in contrast, PBAMs used as subsystems also require connections to their *product variables*.

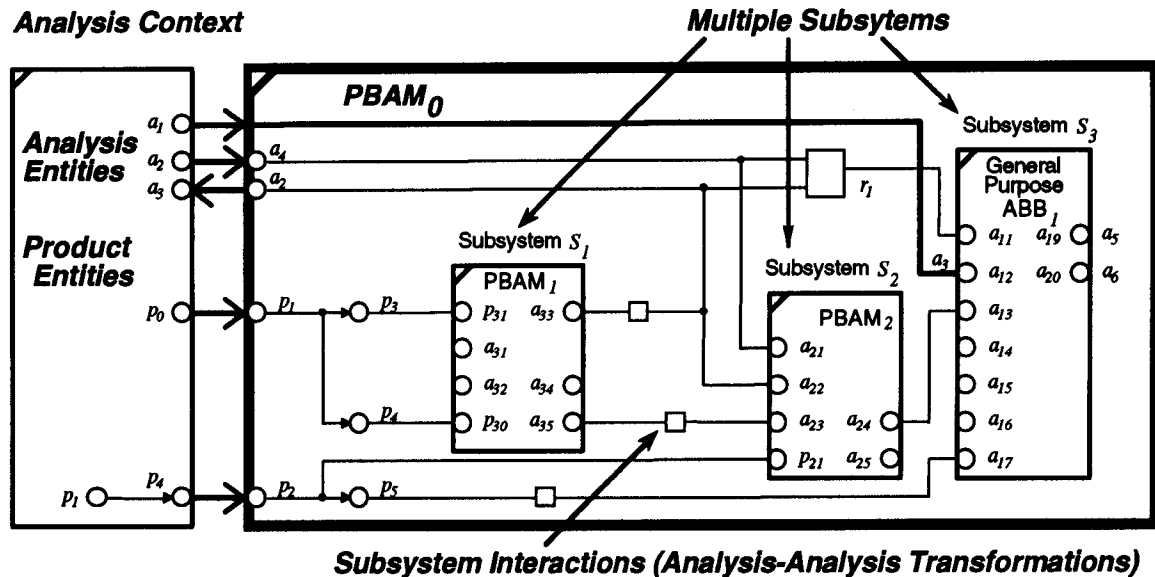


Figure 6.8 Structure of a Complex PBAM

Also note that subsystems can be connected to each other in a typically "my output = your input" fashion (where input/output is relative to which variable is viewed as the output of the outer PBAM). Thus, the emphasis of complex PBAM is usually on such AATs between subsystems rather than on PATs as in a simple PBAM.

EXAMPLE 6.3 Complex PBAM (Two Component Extensional Model)

A simplified analysis model will now be used to illustrate the above complex PBAM concepts. Consider the case where two possibly dissimilar electrical components are placed side by side and subjected to thermal loads. Assume someone¹ is interested in the combined deformation of these two contacting components.

¹ Admittedly it is doubtful that this scenario would be very useful in actual PWA design - it is given here for illustration purposes only.

One possible analysis model is shown in Figure 6.9 where it is assumed that the temperature loads are uniform and that the component undergoes uniform extension (subscripts ci denote component i , $i=1,2$).

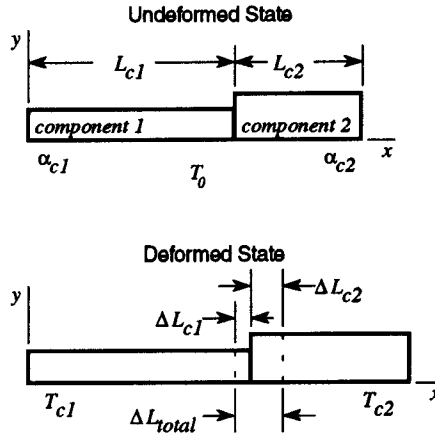


Figure 6.9 Combined Elongation of Two Electrical Components

The following equations would govern the deformation behavior of this analysis model.

$$\Delta L_d = \alpha_d (T_d - T_o) L_d \quad (6.4)$$

$$\Delta L_{c2} = \alpha_{c2} (T_{c2} - T_o) L_{c2} \quad (6.5)$$

$$\Delta L_{total} = \Delta L_{c1} + \Delta L_{c2} \quad (6.6)$$

As one might guess, the PBAM from Example 6.2 (Component Extensional Model) can be used to build a PBAM of this analysis model, as Eqns. 6.4 and 6.5 are present in that simple PBAM. Because the electrical components might be different, the Component Extensional Model is used in each of two subsystems (Component 1 Model and Component 2 Model) in this new PBAM (Figure 6.10) which is called a Two Component Extensional Model.

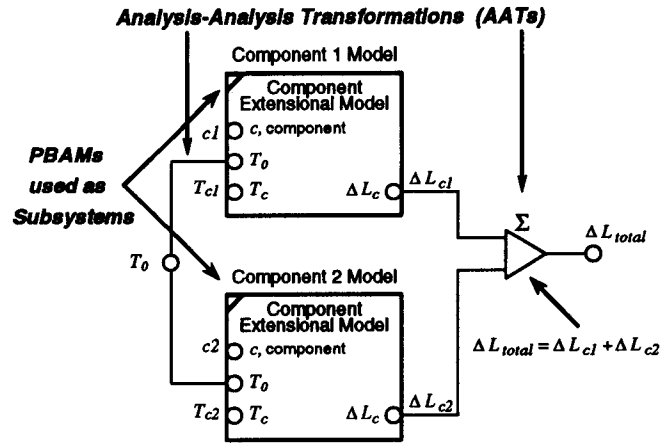


Figure 6.10 Constraint Schematic for Two Component Extensional Model

Because each electrical component is assumed to have the same reference temperature (i.e., $T_0 = T_{0,c1} = T_{0,c2}$), two AATs (equality relations, in this case) are present between the reference temperature variable in each subsystem and the reference temperature variable in the complex PBAM as indicated. Eqn. 6.6 is represented by the summation relation shown, which is also an AAT.

An analysis context uses this PBAM by inputting the product model representation of each component (i.e., subinstances of Electrical Component) into the appropriate product variables in the subsystem (i.e., variables $ci = \text{Component } i \text{ Model.component}$, for $i=1,2$). The analysis context would also input the temperatures T_0 , T_{c1} , and T_{c2} if total elongation, ΔL_{total} , is the desired output.

Figure 6.11 shows one possible subsystem view of this new PBAM, which in turn can be used on other PBAMs. In summary, this relatively easy analysis model has illustrated the complex PBAM concepts of nested PBAMs and AATs.

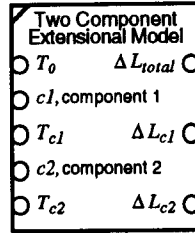


Figure 6.11 Example Subsystem View for Two Component Extensional Model

6.5 PBAM Views

This section defines general views of the PBAM representation, which are actually specialized versions of the ABB views. In several cases, the PBAM view is the same as the corresponding ABB view except that the variables can be divided into product variables and analytical variables.

6.5.1 PBAM Structure

The **PBAM structure** is basically the ABB structure with more distinctions between types of relations and variables. A few definitions will be given first to prepare for these distinctions.

DEFINITION 6.5 A **PBAM partition**, *PP*, is an ABB partition that categorizes variables and relations according to their use in the PBAM structure. These categories are:

product variables, <i>P</i>	PBAM-specific relations, <i>PSR</i>
analytical variables, <i>A</i>	linkages, <i>L</i>
subsystems, <i>SS</i>	

A PBAM partition is used in a PBAM in a similar manner as an ABB partition is used in an ABB - mostly as a primary partition or an option in a PBAM. The above distinctions are made to aide in the development and implementation of PBAMs as well as product models (since product models are used by PBAMs).

DEFINITION 6.6 A **PBAM (product model-based analytical model)** is an ABB that includes linkages between product variables and analytical variables. In a PBAM, the primary partition and all options contained in the option categories are PBAM partitions.

To summarize the general structure of a PBAM, the template in Figure 6.13 is given where the same notation is used as in the preceding chapter for the general structure of an ABB.

```

PBAM
  primary partition,  $PP$ 
    product variables,  $P = \{\text{product variable}_i\}$ 
    analytical variables,  $A = \{\text{analytical variable}_i\}$ 
    subsystems,  $SS = \{\text{analytical model}_i\}$ 
    PBAM-specific relations,  $PSR = \{\text{constraint}_i\}$ 
    linkages,  $L = \{\text{constraint}_i\}$ 
  option categories,  $OC = \{$ 
    option category $i$ 
      common partition $i$ 
      options $i$ ,  $OP_i = \{$ 
        PBAM partition  $j$ 
          product variables,  $P_j = \{\text{product variable}_k\}$ 
          analytical variables,  $A_j = \{\text{analytical variable}_k\}$ 
          subsystems,  $SS_j = \{\text{analytical model}_k\}$ 
          PBAM-specific relations,  $PSR_j = \{\text{constraint}_k\}$ 
          linkages,  $L_j = \{\text{constraint}_k\}$ 
        }
      }
    }
  
```

Figure 6.12 Abbreviated Structure of a PBAM

The PBAM structure serves the same purpose as the ABB structure and is given in general form in Figure 6.13. Again, the same notation used in the ABB structure is used here. Figures 6.14 and 6.15 are example PBAM structures that have been filled in with information specific to the analysis models in Examples 6.2 and 6.3. As with the examples

of general purpose ABBs in the preceding chapter, the Component Extensional Model and Two Component Extensional Model constraint schematics described in Examples 6.2 and 6.3 were used as guidelines to fill in the PBAM structures representing these two analysis models, and the views of other ABBs used as subsystems were consulted. In contrast to general purpose ABBs, the product model also had to be consulted to see what product attributes and standard PATs are available or need to be added to the product model.

6.5.2 Constraint Schematic

The same constraint schematic notation can be used for PBAMs as is used for ABBs. (Note that no new constraint schematic notation was introduced in this chapter). Product variables, PATs, and AATs fit within the existing constraint schematic framework as they are just special types of variables and relations.

The usefulness of this view is hopefully more apparent with PBAMs since they add product information and higher level interactions between subsystems. It helps one graphically visualize the relations and linkages between product data, analysis data, and subsystems at an abstracted level compared to a constraint graph. If constraint schematics and subsystem views already exist or are readily developed for the subsystems a new PBAM will use, one could develop this new PBAM to a large degree by directly drawing its constraint schematic. The schematic then could be used to fill in most of the PBAM structure. The visualization advantages also make this view a helpful one for both implementors and users of the PBAM.

6.5.3 Object Relationship Diagram

The primary difference between PBAM and ABB object relationship diagrams is that ones for PBAMs have attributes that are product model entities.

PBAM name
Superclass: PBAM super class

Option Categories

q. option category ...
[q.i] option ...
Product Variables
[q.k] product variable ...
analytical variable* ...

Product Variables
[q.i] option ...
[q.k] product variable ...
analytical variable ...*

Product variables
[q.k] product variable ...
analytical variable ...*

Other Analytical Variables
 $[q,k]$ analytical variable...

subsystems
[q.k] subsystem...
subsystem variable ...

Semantic Linkages	Subsystem Variable
<u>PBAM Variable</u>	<i>name ...</i>
$[q.k]$	<i>name ...</i>

PBAM-Specific Relations
 $[q.k]$ *t. relation* ...

Other Linkages
 $[q.k]$ *relation* $r(v_i, \dots)$...

*** Analytical variables resulting from PATs.**

Figure 6.13 General PBAM Structure

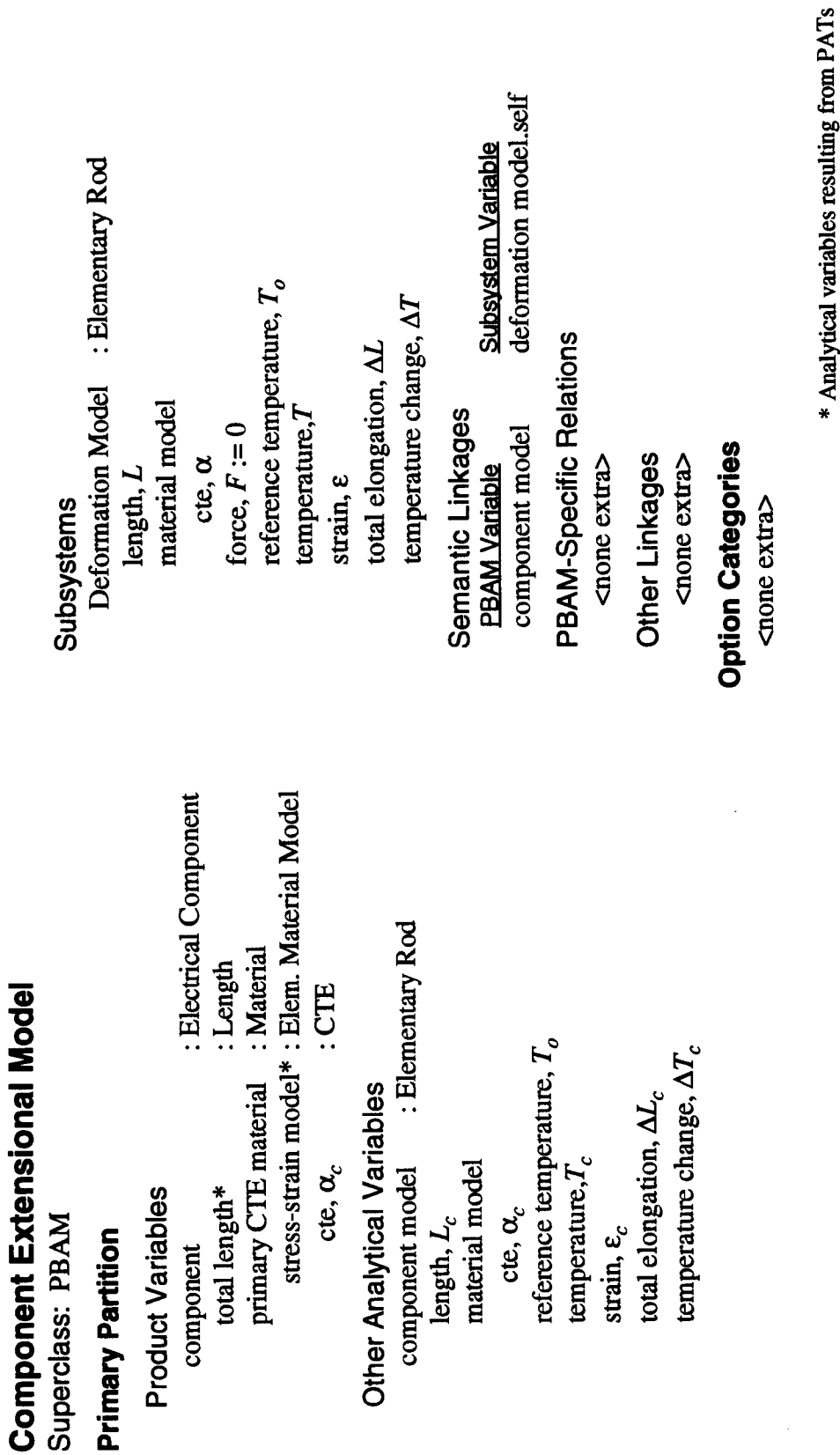


Figure 6.14 PBAM Structure of Component Extensional Model

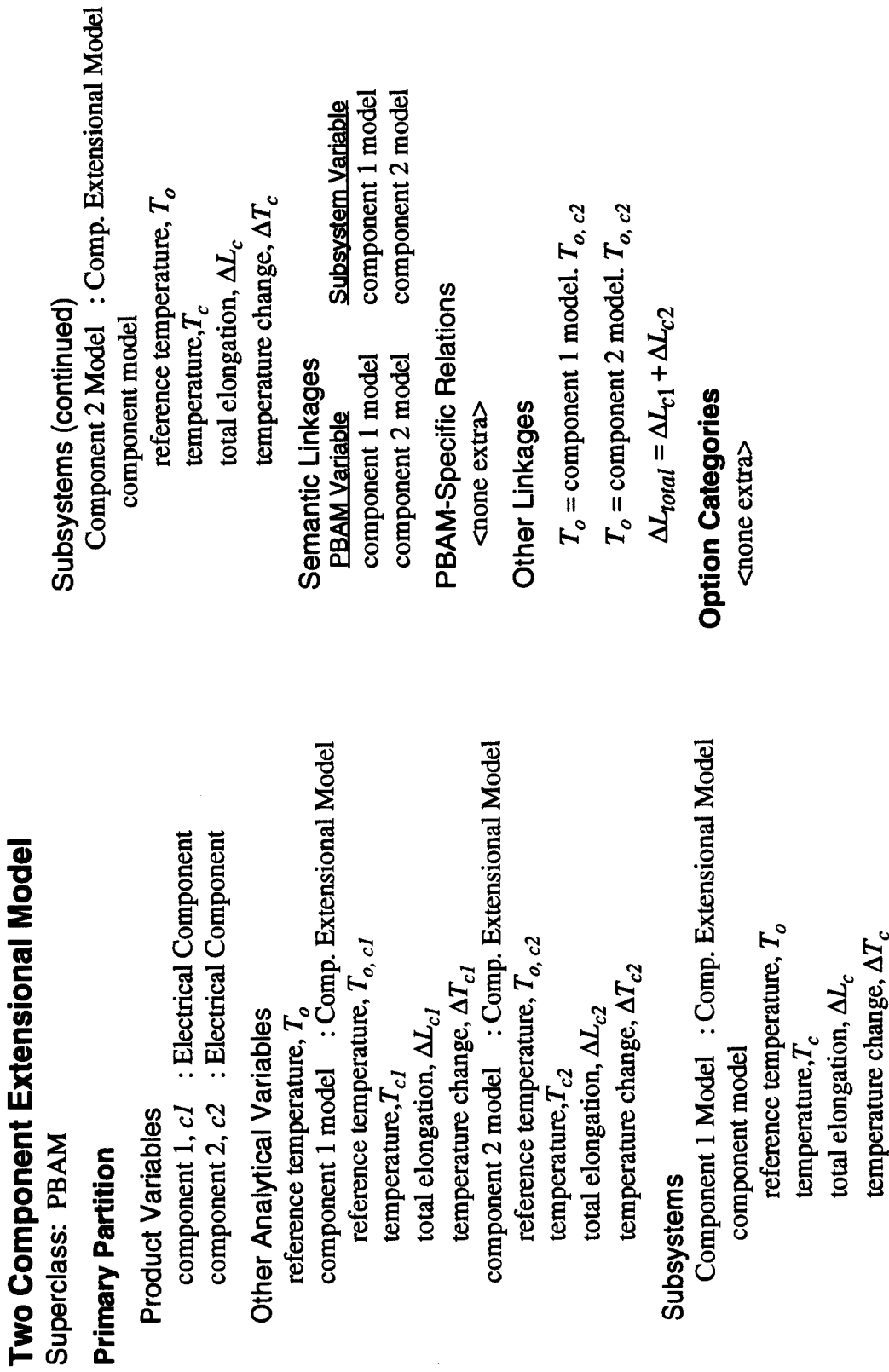


Figure 6.15 PBAM Structure of Two Component Extensional Model

6.5.4 Subsystems

There are no syntactic differences between subsystem views of PBAMs versus ABBs. The primary difference is semantic in that the variables shown on a subsystem can be categorized as either product variables or analytical variables. Though possible, a subsystem view that shows only analytical variables probably defeats the purpose of a PBAM as apparently no interaction with the product model would be intended in this case.

6.5.5 Extended Constraint Graphs

Again, the primary difference is in the semantics of the distinction between product variables and analytical variables.

6.5.6 Input/Output Tables

Same as above.

6.5.7 Instance Views

Detailed instance views were used in Figures 6.4 and 6.8 to show the general forms of simple and complex PBAMs, respectively. Similar to the above structural ABB/PBAM views, the primary difference versus an ABB instance view is the categorization of the variables that the analysis context connects to the PBAM. They are distinguished as either **analysis entities** or **product entities** depending on what type of PBAM variable they connect to.

6.6 Summary

This chapter reviewed product model concepts as a necessary foundation before defining the product model-based analytical model (PBAM) representation. Then the PBAM representation was defined as a special case of the ABB representation which includes linkages between general purpose analysis models and a specific type of product model.

Specifically, in the PBAM representation, variables are categorized as either product variables or analytical variables, and linkages between these two groups are called product-analysis transformations (PATs). Because such transformations are represented as relations, they can be included in the constraint schematic of a PBAM.

An example simple PBAM (a PBAM with only one subsystem that is a general purpose ABB) was given that utilized a general purpose ABB from the previous chapter. Another example showed how this simple PBAM could be used as a subsystem in a complex PBAM (a PBAM with one or more subsystems, which can be any type of ABB including other PBAMs).

Finally, a comparison was given between the ABB and PBAM representation that showed how all ABB views except the ABB structure are directly applicable to PBAMs. The primary difference between the ABB structure and the PBAM structure is the distinction between product and analytical variables - a distinction which is helpful for development and implementation purposes as discussed in the next two chapters.

CHAPTER 7

PRELIMINARY PBAM DEVELOPMENT GUIDELINES

This chapter discusses how representing a given analysis model using PBAMs involves using filling in the PBAM structure related views defined in Chapters 5 and 6 with the information specific to that analysis model. It gives preliminary PBAM Development Guidelines for this process, which is called "*developing a PBAM*". "*Implementing a PBAM*", discussed in the next chapter, is the process of casting a developed PBAM into a computerized form in a specific computing environment.

7.1 Analysis Model Descriptions

As PBAMs are intended primarily to represent analysis models in order to automate *routine* analysis, it is assumed descriptions of the analysis models exist in some form (i.e., a analysis model must exist to some degree in order to develop a PBAM to represent it). Therefore, given the PBAM structure defined in the preceding section, one can view developing PBAMs of specific analysis models as mapping from the typically loosely structured descriptions of those analysis models into the structured PBAM representations. Appendix B contains copies of the case study analysis model descriptions as examples. Common sources of such descriptions are listed in Table 7.1.

Table 7.1 Common Sources of Analysis Model Descriptions

- | | |
|-----------------------------|----------------------------------|
| 1. Journals | 5. Corporate technical memoranda |
| 2. Conference proceedings | 6. Unpublished notes |
| 3. Textbooks | 7. CAE tool input files |
| 4. Handbooks, Topical books | 8. Computer programs |

These descriptions typically include minimal documentation of product-analysis transformations and little product data. They can lack even analysis-oriented information to the degree that reproducing example results is difficult.

7.2 PBAM Development Steps

Each step in the PBAM Development is now explained.

Step 1: Identify the purpose that the analysis model is to serve. This purpose could first be stated in relatively non-technical terms such as "Determine if the member will break." Next determine the technical factors that need to be considered to fulfill such a purpose, e.g., "check the von Mises stresses under maximum loads," and "check if fatigue is an issue." Also identify variations that need to be included in the model, such as the different types of loads.

The necessary items to continue this process include a description of relevant product models and "data sheets" for the potential ABBs that can potentially be used to construct the PBAM.

Step 2: Identify analysis models to fulfill the purpose identified in Step 1. One can develop new analysis models or find existing analysis models in the sources identified above (Table 7.1). Papers describing the case study analysis models are included in Appendix B for reference when going through Chapter 9.

Step 3: Define the major steps of the analysis models. Develop a high-level top-down view of the major analysis model steps for the case where only one of each type of variation identified in Step 1 is considered. A tree/network diagram such as Figure 7.1 (reproduced from Chapter 9¹) may be helpful, or an IDEF₀ process model could be developed.

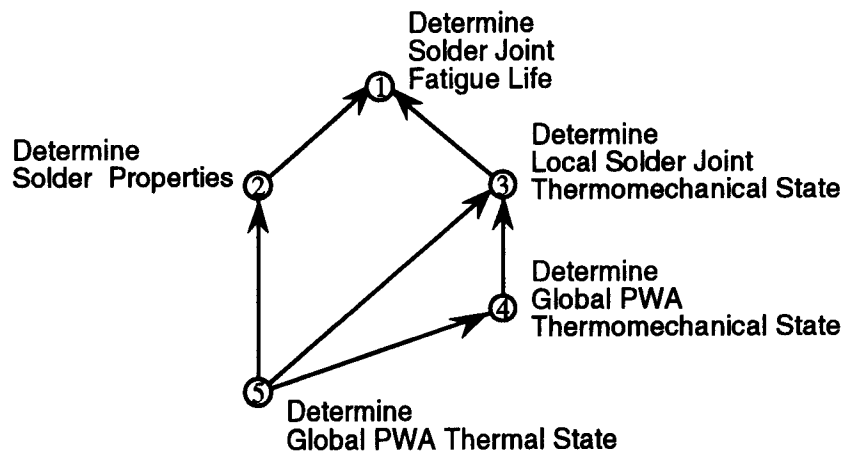


Figure 7.1 Identification of Major Steps in Solder Joint Fatigue Analysis

Upon the completion of this step, one also should be able to "manually" walk through the analysis model(s) and find solutions using representative datasets. "Manually" means that, at a minimum, the analyst manually provides input to the appropriate tools and manually exchanges data among these tools as necessary (e.g., manually using a calculator, creating and running a finite element analysis model, and exchanging information among such tools). This exercise also will help better understand the analysis

¹ Unfortunately, examples from Chapter 9 (Case Studies) are referenced in this chapter before the reader has read about them. Where possible, examples are also given from Chapters 5 and 6. Creating examples before this chapter that exercise advanced aspects of the PBAM representation (such as options and subsystem substitution) was thought to require too much duplication of effort since good examples exist in the case studies. Therefore, the reader is encouraged to review this chapter again after gaining familiarity with the case studies in Chapter 9.

model(s) themselves and provide some reference datasets to check the PBAM implementation(s) later. "If you can't do it without a computer, don't expect to do it with a computer" needs to be kept in mind here.

Note that Steps 1, 2 and 3 of this PBAM development process may have already been done if the analysis model of interest is truly a routine analysis model (i.e., it has been used repeatedly). The steps covered thus far are intended to create an informal analysis model description if one does not already exist.

The description of Example 6.2 (Component Extensional Model) and Example 6.3 (Two Component Extensional Model) illustrate simple such descriptions. The description of the case study analysis models in Section 9.2 can serve as a guideline for creating descriptions of more complicated analysis models if necessary. That section was achieved by extracting the relevant information from the papers in Appendix B and combining them in one place. Some gaps were filled in, and minor extensions were added as discussed in that section (e.g., applying the power cycling loading option to the Plane Strain Model).

Perhaps the most useful thing to do at this point is *identify and organize the analysis equations* used under each major step of the analysis model. Use product-specific notation to identify variables at this point (e.g., subscripts Eqns. 6.2, and 9.2 indicating *component* temperature and *solder joint* height respectively). Define relations with finite-element based solutions according to their input parameters (e.g., 9.11 and 9.12).

Step 4 Determine if existing PBAMs and GPABBs can be used as-is or readily adapted to supply the relations identified in Step 3. If so, note such cases by placing them as subsystems in an initial constraint schematic of the overall analysis model. Example 6.1 and 6.2 exemplify this point (e.g., Eqn. 6.2 was determined to be available in the

Elementary Rod GPABB). The scope of the existing available GPABB and PBAM libraries will dictate how much can be achieved in this step by ABB re-use.

Step 5 Decompose remaining major steps identified in Step 3 until the skeleton constraint schematic is reasonably well defined. Each vertex that is not a subsystem will probably be a variable or relation that is contained in the original description of the analysis model at this point. At this point incomplete and/or missing variables and relations may have to be refined by the analyst. Relations with no closed form representation (such as those requiring a finite element analysis) can be represented by their variables and the relation symbol (S2).

Step 6 Collect the relations outside any subsystems into "natural" groupings so that existing ABBs can be used to package these relations and variables if possible. One example of natural groupings are given in Example 5.3 (Interconnected Rods System) where the geometric, material, and load variables are related to the body to which they semantically belong. Further discussion regarding natural groupings can be found in Appendix F. If necessary, develop new GPABBs that are needed for this particular analysis model and that could potentially be used for others as well. If appropriate, package both the existing and newly identified primitives into new analytical systems. If any analysis relations still remain in the scope of the developing PBAM (i.e., outside of all subsystems), these relations probably can be classified as PBAM-specific relations that are AATs (defined in Chapter 6).

Step 7 Add product variables and product-analysis transformations into the list of equations developed in Step 3. Determine if a PAT is standard over the definition given in

Chapter 6 (e.g., it would be commonly useful inside other PBAMs). Update the constraint schematic with these PATs so that most of the dangling analytical variables are provided with values derived from the product model. The remaining dangling analytical variables will most likely be boundary conditions and performance variables which can be left as analytical variables in the PBAM or move into a subsystem..

Identify which product variables in the constraint graph already exist in the product model and which ones need to be added. Determine how to package and place any new product variables into the product model, as well as any new standard PATs.

Recursively add parent product variables (with respect to the part-of relationship) to each product variable until one of the following conditions is met:

- 1) All required product variables have a single common parent product variable (this would be the eldest product variable and the only patriarch (Chapter 6). The component variable, *c*, in Figure 6.6 of Example 6.2 is such a variable.
- 2) All remaining "parentless" product variables meet one of the following conditions
 - a) The product variable is a patriarchal variable (Chapter 6).
 - b) All product variables not meeting condition (a) are not part of any common assembly (therefore they are distinct products). One case where this situation would arise is in modeling the interaction of products contained in some common media, such as modeling two ships/cars crashing together on the ocean/on a road. Here, there would be three patriarchal "product" variables that meet this condition (b): two ships/cars and the ocean/road. It could be argued that the ocean and road are the "common assembly" in these cases.

All resulting product variables that are roots in the product variable part-of trees are categorized as patriarchal product variables. Upon completion of this step all the

variables and relations should fit into one of the categories in PBAM partition, and a constraint schematic that is pretty close to final form should emerge.

Step 8 Repeat Steps 3 through 7 for other analysis model variations to identify options and potential other PBAMs.

Generally variations that cause a change in constraint graph topology should be represented individually as a PBAM option. Similar variations that are mutually exclusive can be grouped together to form options in a PBAM option category.

If a variation does not cause a major change in constraint graph topology, then it need not be cast as an option, but may be considered a generalization of a relation. For example, the product-analysis transformation used in the Component Occurrence Deformation Model PBAM called approximate maximum inter-solder joint distance depends on the type of component being analyzed. Consequently, the generalized relation has component type as a variable.

Another example of when to make option categories occurs in the Solder Joint Thermomechanical Fatigue (SJTF) Model PBAM (Appendix G); including warpage and power cycling effects cause topology changes within the scope of the PBAM. Therefore, these subsystems were identified and associated option categories were added.

Note that while the strain model option does not change the PBAM scope topology, many changes occur in the inner strain model topology (compare constraint schematics of the Extensional Model versus the Plane Strain Model). Originally when these PBAMs were being developed, there were only two classes of PBAMs (Extensional Solder Joint Fatigue Model and Plane Strain Solder Joint Fatigue Model). Due to the high degree of commonality and the many topological differences, it seemed better to make separate

PBAMs for the strain calculation step, and add a subsystem substitution option to the SJTF Model result in the present form seen in Appendix G.

Presently, there are no hard and fast rules on when the major steps in an analysis model should be broken into separate PBAMs. One guideline is to do so if the constraint schematic gets too complicated and there are natural boundaries for such a division (e.g., create a PBAM to handle each major step in the analysis). Anticipated degree of re-use is another factor to consider, e.g., how many other PBAMs for other applications could use an Extensional Model PBAM as a subsystem.

Upon completion of all eight steps the analysis model and its variations should be represented as a collection of PBAMs and other ABBs. Other PBAM views besides the PBAM structure and constraint schematics that can aide this development process include object relationship diagrams, extended constraint graphs, and subsystem views.

Better PBAM Development Guidelines can be expected if PBAMs become more widely used. Chapter 10 discusses the related topic of PBAM equivalence in terms of potential benefits of constraint graph theory.

7.3 Useful Development Skills

Useful skills for PBAM developers (there need not be necessarily a single person with all these capabilities) include:

- Product modeling capability.
- Analysis model understanding, including where product variables come from, and what kind of tools are appropriate for solving the analysis model.
- Familiarity with ABB libraries.
- Basic object-oriented concepts.
- Basic understanding of constraint graphs.

7.4 Discussion and Summary

This chapter defined developing as PBAM as the process of using a description of an analysis model to fill in the structural views of a PBAM. Common sources of such analysis model descriptions were given (e.g., journal papers and handbooks). Eight steps in the initial PBAM Develop Guidelines were explained with reference to the examples in Chapter 5 and 6, as well the case studies in Chapter 9. Finally, basic PBAM development skills that would prove useful were given. In conclusion, it is felt that these guidelines and the examples given in Chapters 5, 6, and 9 (and related Appendices F and G) are a good start for people to learn how to develop PBAMs for their own applications.

CHAPTER 8

PRELIMINARY PBAM IMPLEMENTATION GUIDELINES

8.1 Philosophy

It is important to note that the PBAM representation itself (as defined by the ABB¹ representation in Chapter 5 and the PBAM structure in Chapter 6) is for the most part independent of how an ABB will be implemented. However, it must be admitted that the ABB operations in Chapter 5 were defined with a definite bent toward implementation as constraints with strength hierarchies [Freemen-Benson, et al., 1990]. At one extreme, the ABB representation potentially could be used simply to document how a particular analysis is done. The documentation could be used to guide someone in "manually" performing each analysis calculation and the information interchange among calculations.

However, to experience rapid and flexible analysis capabilities, some degree of CAD framework integration such as that given in Appendix D could be used (the more, the better). Thus, the available CAE/CAD tools do not necessarily effect the development of a ABB; rather they effect how much a ABB can be implemented.

As with other representations, ABBs could be implemented in a variety of ways (from machine code to object-oriented programming), but the most natural implementation form appears to be as objects in object-oriented programming. Note that the object *representation* is distinguished here from an object-oriented programming

¹ As PBAMs are a specialization of ABBs, the term ABB will be used in this chapter unless a distinction is necessary.

(OOP) *implementation* since non-OO languages also could be used to implement objects (probably with more difficulty). Actual implementation of the object representation is most naturally achieved via an OOP language (e.g., Smalltalk or C++).

Many of the relations in ABBs can be implemented quite naturally as constraints within such an object-oriented environment. Maloney [1991] gives general guidelines on implementation of various applications (e.g., user interfaces) using constraints. Guidelines specifically for implementing ABBs using constraints and objects are the focus of this chapter (with particular reference to a Smalltalk implementation using DeltaBlue/ThingLab II).

8.2 Implementing Structural Aspects Using Object-Oriented Programming

The ABB data structure can be mapped very closely into an object-oriented language, as is true with the other the product entities and ABBs given in Appendix E and F, respectively. In a nutshell, entities in the EXPRESS-G models become classes, and their attributes become instance variables (Table 8.1). Object-oriented concepts are expressed here using terminology from Appendix A. Attributes that are sets are easily implemented in an object-oriented programming (OOP) environment, such as Smalltalk [ParcPlace, 1992] which has an extensive class library.

Table 8.1 Mapping Analysis Models into the Object Representation

<u>Analysis Model Characteristic</u>	<u>Object Representation Feature</u>
• variables	• attributes
• relations	• constraints and methods
• general & special cases (is-a relationship)	• class hierarchy
• bodies & systems (part-of relationship)	• attributes

The ABB structure of an analysis model can be expressed in other (partially) equivalent forms. For example, the STEP EXPRESS information modeling language [ISO 10303-11, 1992] can be used to define the basic structure of an ABB. The new type of ABB can be designated as an ancestor class of the general ABB class. All of the eldest variables (even in each option) can be defined as attributes of the basic ABB class (though not all attributes will be populated). The ABB-specific relations and linkages can be defined using EXPRESS WHERE rules if necessary functions are defined elsewhere in EXPRESS.

Using this approach, the DERIVED attribute construct in EXPRESS appears unnecessary. However, the present version of EXPRESS does not support specifying the actual *values* of the attributes, so it is awkward to define which ABB-specific relations and linkage belong to which option. The STEP standards effort is currently developing an extension specifically for expressing instances, EXPRESS-I [ISO 10303-12], that may help alleviate this problem.

In spite of this lack of instance expression, a (partial) EXPRESS definition of an ABB is still potentially useful for at least two reasons:

1. Formal mappings from an EXPRESS definition into various implementation forms exist and/or are being developed (e.g., into SQL commands or C++/Smalltalk class definitions) [ISO 10303-22]. Some tools exist which automate this mapping process to some degree, e.g., STEP Programmers Tool Kit [STI, 1993]. Wilson [1993] highlights a number of such tools and related applications.
2. A formal mapping from *instances* of entities defined in EXPRESS into a neutral exchange form exists [ISO 10303-21].

Therefore, given an ABB representation of a specific analysis model, its implementation can be automated to the degree that such mappings and tools exist. However, by reviewing Wilson's list of tools referenced above, it is evident that the STEP effort to date has focused only on mappings to achieve implementation of data structure and exchange of instances. Automated implementation of the EXPRESS WHERE rule and FUNCTION constructs [ISO 10303-11] apparently has not been achieved yet.

Partial EXPRESS and EXPRESS-G definitions of the general PBAM structure are given in Appendix G subject to the above limitations. As Appendix A explains, EXPRESS-G is a graphical subset of the (textual) EXPRESS information modeling language. Some EXPRESS constructs, including WHERE rules and FUNCTIONS, are not represented in EXPRESS-G.

In summary, the ABB structure, a textual definition of a particular type of ABB that represents a specific analysis model, can be depicted partially in standard textual forms like EXPRESS (as well as standard graphical forms like EXPRESS-G as is done in a object relationship diagram as defined in Chapter 5). Such forms are advantageous when mappings exist to implement automatically at least some of the structure of objects in general, and ABB in particular.

8.3 Implementing Relations Using Constraints

Implementing the relations belonging to ABBs and to the product model is less straight forward, but is greatly eased by an existing class library of general purpose constraints, such as those in ThingLabII/DeltaBlue [Maloney, 1991]. Such classes are the same as any other class in Smalltalk.

A relatively simple example implementation of the relation between T_{sj} , T_o , T_s and T_c in the Solder Joint Thermomechanical Fatigue Model PBAM (Eqn. 9.3 given here as Eqn. 8.1) is given in Figure 8.1.

$$\bar{T}_{sj} = \frac{1}{4}(2T_o + T_c + T_s) \quad (8.1)$$

The `var:` statements specify to which PBAM variable/subvariable the local constraint symbols (`to`, `ts`, `tc`, `tsj`) are bound to and show one way that equality relations are formed (`self` refers to the PBAM instance itself that uses this constraint). Actual equality constraints are used in other cases. Thus, this example shows how formula-based relations are easily implemented.

```

c1 := Constraint names: #(
  tsj
  tc
  ts
  to)
methods: #(
  'tsj := 0.25*(tc + ts + (2*to))'
  'tc := (4*tsj) - ts - (2*to)'
  'ts := (4*tsj) - tc - (2*to)'
  'to := 0.5*((4*tsj) - tc - ts)' ).

c1
var:(self meanCyclicSolderJointTemperatureVar) "tsj"
var:(self strainModel componentModel temperatureVar) "tc"
var:(self strainModel pwbModel temperatureVar) "ts"
var:(self strainModel referenceTemperatureVar) "to"
strength: #required

```

Figure 8.1 Exemplar Implementation of a Constraint

One must be careful about how constraints are connected to each other in developing operations like those given in Chapter 5. Also, the analysis context must not haphazardly change constraint and variable strengths, or else unintended variables may change, and erroneous results will be obtained. For example, even if a variable has been

specified as an output, someone can still try to input a value into it. However, the value will not remain what was input into, as the constraint solver will change it to satisfy other constraints on that variable. A more subtle error occurs if no output variable is specified (or if more than one variable is). In this case it is generally unpredictable which variable values will change and which will remain the same.

The implementation of relations requiring finite element-based solutions is just as easy as formula-based ones from a purely constraint point of view. Relations among variables in the analysis can be captured in analytical system objects (as in the Plane Strain Bodies System in Appendix F). In this research parameterized ANSYS PREP7² models [SASI, 1990] for this generic system were developed (that could be used by applications other than solder joint fatigue). With this approach, each PREP7 parameter (including mesh density parameters, number of load steps, etc. if desired) could be related to variables in the constraint (but not necessarily with a 1:1 correspondence).

A Smalltalk method would be referenced in the constraint creation code for each desired output possibility (as in Figure 8.1 above, e.g. for t_{sj}). Each method can be implemented in the analytical system class that requires a finite element solution. Basically, a constraint calls the method that is needed to obtain the desired constraint output. That method transforms the analytical system variables into the parameters needed by the ANSYS PREP7 file (or equivalent). Then that file is automatically created and submitted for solution. After the FEA solver is finished, the results come back to the method in the analytical system. The method then passes the results to the constraint which called it. Thus, the constraint views a finite-element relation as it would any other

² ANSYS PREP7 models are typically finite element models that are at a higher level of intent than a file just containing elements, nodes, and nodal loads. For example, one can specify geometric regions and how they should be automatically meshed, versus explicitly specifying every node and element in the region.

relation. Practically, however, one should prevent the relation from reacting to every change in the constraint graph. This can be done, for example, by relaxing the relation until all its inputs have settled.

8.4 Summary

This chapter has highlighted initial guidelines for implementing ABBs and PBAMs using constraints and objects in an object-oriented programming language. How to map from the ABB/PBAM structure into an object-oriented programming environment was included, as well as how to represent relations with finite element-based solutions as constraints was included.

PART III VALIDATING THE PBAM REPRESENTATION

CHAPTER 9

PWA THERMOMECHANICAL ANALYSIS CASE STUDIES¹

This chapter shows how PBAMs were developed and implemented for representative analysis models selected to evaluate and illustrate the PBAM representation. Section 9.1 gives a background on solder joint reliability in general to give a context for the specific analysis models used in the case study, which are described in detail in Section 9.2. Section 9.3 describes how PBAMs were developed to represent these analysis models, and Section 9.4 discusses implementation of these PBAMs, while Section 9.5 gives test runs with for design verification and design synthesis scenarios. The conceptual development of a PBAM for PWA warpage analysis and its potential use in a global/local analysis scenario is described in Section 9.6. Section 9.7 contains general discussion with respect to the case study results.

9.1 Solder Joint Reliability Under Thermomechanical Loads

The advent of surface mount technology (SMT) has brought many benefits and challenges to the printed wiring assembly (PWA) industry. Primarily, surface mount components require less space on the surface of the printed wiring board (PWB) and also do not require holes for mounting (see Figure 9.1).

¹ A condensed earlier version of this Chapter has been published as [Peak and Fulton, 1993b].

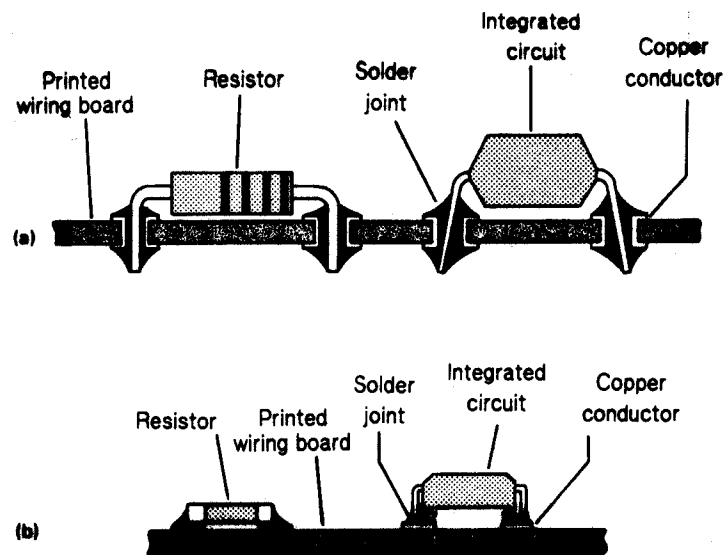


Figure 9.1 Comparison of a) Through Hole and b) Surface Mount Technology
[Hinch, 1988, p.4]

As further noted by Hinch [1988, pg. 14-18], challenges brought on by SMT include the needs for greater accuracy in assembly (due to smaller sizes) and more attention to thermomechanical reliability issues. The basic reasons that thermomechanical reliability has become more of an issue are as follows:

1. Power densities have increased with decreasing package size, which tends to increase component and PWB temperatures.
2. Surface mount components tend to be stiffer with respect to their mounting surfaces than their through hole counter parts.
3. Whereas though hole components are mechanically attached to the PWB primarily through relatively robust wire leads, SMT relies on solder joints (and sometimes adhesive) for their mechanical connections.

4. Components, PWB and solder (and adhesive, if used) all are made of different materials that have different material properties and (in particular, different CTEs). Therefore, as a PWA undergoes thermal excursions (which can be significant due to reason 1.) the component and PWB expand different amounts resulting in an often large finite strain on the solder joint. With repeated cycling, one or more solder joints begins to crack, causing high resistances or electrical disconnections which usually lead to product failure.

Though these are causes of solder joint failure due to thermal loading, it should be noted that PWAs encounter other types of loading that cause solder joint failures, including vibration, corrosion, mechanical flexure. [Lau, 1991].

As preventing product failure is a concern to most manufacturers of electronic products, there has continued to be a great deal of activity in industry and academia to improve solder joint reliability. Several conferences held annually/regularly devote entire sessions to this topic (e.g., IEEE Electronic Components & Technology Conference, ASME Winter Annual Meeting, ASME Electronic Packaging) and several journals report on research programs in this area (*ASME Journal of Electronic Packaging*, *IEEE CHMT*.) At least one book has been devoted to this subject to date, [Lau, 1991], and another is forthcoming, [Lau, 1993]. Therefore, one can see that thermomechanical issues in electronic packaging, in general, and solder joint fatigue on PWAs in particular, are relevant and timely research areas in their own right.

However, in spite of its importance, to date there is no known, commercially available, computer-aided tool that PWA designers can use to evaluate the solder joint reliability of their designs. One prototype system has been developed at the University of Maryland (CALCE), which has reported use in industry [Nilson, 1991]. This prototype

focuses on the solder joint reliability problem itself rather than on general ways to represent a variety of analysis models. Therefore, case studies from this area not only serve to illustrate and evaluate the PBAM approach, but also demonstrate applicability to a relevant industrial need.

9.2 Case Study Solder Joint Fatigue Analysis Models

Nomenclature

\bar{N}_f	average cycles to failure
$\Delta\epsilon^p$	plastic cyclic strain range
c	fatigue ductility exponent
ϵ_f'	fatigue ductility coefficient
\bar{T}	mean cyclic temperature, (°C)
f	load frequency, (cycles/day, $1 \leq f \leq 1000$)
\bar{T}_{sj}	mean cyclic solder joint temperature (°C)
$\Delta\gamma_{sj}$	solder joint shear strain range
F	adjustment factor
$\Delta(\alpha\Delta T)$	steady state thermal expansion mismatch
T_0	reference temperature
T_{ss}	steady state temperature
L	length
h	height
E	Young's modulus
ν	Poisson's ratio
α	coefficient of thermal expansion (CTE)
σ_Y	yield stress
λ	strain hardening coefficient
ω_c	component occurrence ²
Ω_c	set of component occurrences, $\{\omega_c\}$

² The term **component occurrence** means the usage of a component at a specific physical location in a PWA [Peak and Fulton, 1992b]. It refers to a component-solder joint-PWB assembly. **Component** refers only to a device of a given part number which may occur many times on a given PWA. The unique identifier for an occurrence is a reference designator (e.g. R110) versus a part number (e.g. PN 99120) for a component .

d	mesh density
n	number of load steps
a	load yield factor
e	convergence criteria

Subscripts

pwa	printed wiring assembly (PWA)
pwb	(bare) printed wiring board (PWB)
c, s, sj	component, substrate/PWB, solder joint (e.g., E_c, E_s, E_{sj})

This section overviews how to analyze solder joint reliability using two analysis models chosen from the literature that were used as case studies in this research. *The emphasis of this research is on general methods for representing such analysis models rather than on developing the analysis models themselves.* Copies of the actual papers used are included in Appendix B for reference.

From a top-down viewpoint, the major steps required to predict the solder joint fatigue life for a given component occurrence are shown in Figure 9.2. Each step will now be discussed along with how it is carried out in the case study analysis models.

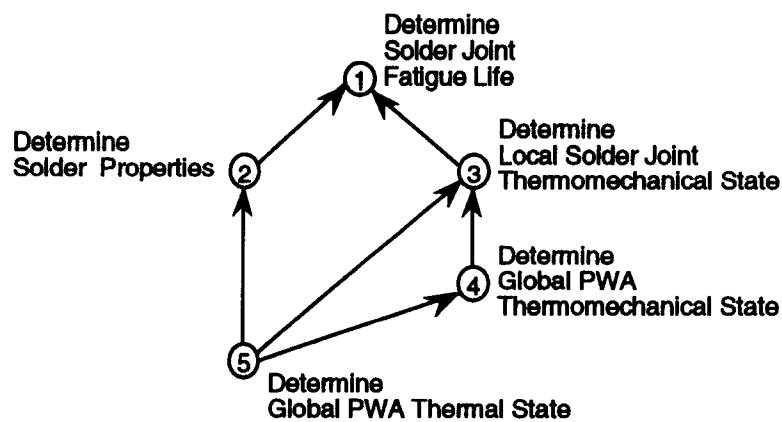


Figure 9.2 Major Steps in Solder Joint Fatigue Analysis

1. Determine Solder Joint Fatigue Life

The first part of this step is deciding what fatigue model to use. As solder is a material with a relatively low melting point, its behavior is characterized by a low yield stress (~5000 psi at 25°C) and by creep under relatively small loads. Therefore, Engelmaier [1983, 1989] uses the below modified Coffin-Manson relation for low cycle fatigue [Lau, 1991, p. 391; Dieter 1983, pp. 273-275] where the exponent, c , is frequency and temperature dependent (Step 2).

$$\bar{N}_f = 1/2 \left(\frac{\Delta \epsilon^p}{2 \epsilon_f} \right)^{1/c} \quad (9.1)$$

The strain range, $\Delta \epsilon^p$, must be found from the structure undergoing thermal loading (Step 3). Other fatigue models have been proposed which are usually more complicated and require different/further information (e.g., strain range partitioning includes the effects of varying load wave forms [Kilinski, et al. in Lau, 1991, p. 393]). Thus, this step drives what other analysis steps must be performed.

It is important to note that fatigue life prediction is a complex process requiring knowledge about many factors [Solomon in Lau, 1991, p. 446]. These factors may be difficult to determine precisely for each PWA being designed. A model that performs well under one set of circumstances may not give accurate results under others. Furthermore, one must be aware of the statistical nature of fatigue failures as considered in the *figures of merit* technique developed by Clech, Engelmaier and co-workers [Engelmaier, 1989]. Therefore one must use fatigue life predictions resulting from relations like the above with caution.

2. Determine Solder Properties

Engelmaier [1983, 1989] developed the following relation (applicable only to 60% Sn - 40% Pb solder and eutectic solder) for input into the above Coffin-Manson relation:

$$c = -0.442 - 0.0006\bar{T} + 0.0174\ln(1 + f) \quad (9.2)$$

Here \bar{T} has units °C and the load frequency, f , has units cycles/day. The load frequency range is $1 \leq f \leq 1000$, where the assumption is made that load frequencies less than 1 cycle/day do not further decrease cyclic life. Note that some characterization of the thermal loads is needed, which Engelmaier provides by the following relation:

$$\bar{T} = \bar{T}_{sj} = \frac{1}{4}(2T_o + T_c + T_s) \quad (9.3)$$

The other material property is assumed to be constant for solder, $\epsilon'_f \approx 0.325$.

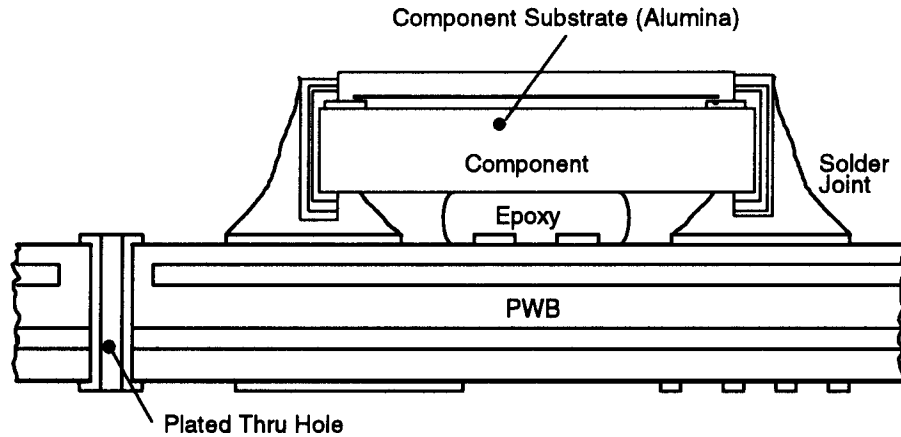


Figure 9.3 Cross Section of a Multilayer PWB/Component Assembly

3. Determine Local Solder Joint Thermomechanical State

As shown in Figure 9.3, the component and board are themselves complex assemblies of different materials. Therefore, one must decide how much of this detail to include in the

analysis model. Figure 9.4 gives some modeling dimensions along with typical example values. Decisions such as which bodies to include and what types of loads to consider are made at the analysis system level. Within each major body in the analysis system, the level of geometric, material, and behavior detail to be considered must be determined. Similar decisions must be made for each sub-body/region within a body if it is to be further decomposed. Finally, some solution method parameters (e.g., for nonlinear finite element analysis) are listed to emphasize that the same analysis model can have different final forms for a given solution method (and, thus, possibly different results).

- Bodies: component, solder joints, PWB, epoxy
- Geometry Model: 1D, rectangular, detailed 2D, detailed 3D
- Material Model: linear elastic, bilinear elasto-plastic, viscoplastic;
temperature/strain rate dependent; isotropic, orthotropic
- Behavior Model: shear body, rod, beam, plane strain, plane stress, plate, shell, 3D
continuum
- Sub-bodies (PWB layers and features, component leads)
- Internal (inter-body) Boundary Conditions: no slip
- External Boundary Conditions/Loads: temperatures, warpage effects
- Load Types
 - Time Variation: initial, transient, steady state
 - Space Variation: uniform, discrete, distributed
- Solution Method Parameters: mesh density, number of load steps, convergence criteria

Figure 9.4 Example Strain Model Variations

Figure 9.5 gives some dimensions along which the PWA itself can vary. Different types of components can require different types of analysis model (e.g., leads on leaded components may need to be modeled). Thus, analysis model representations should support such *product variations* that could impact analysis results.

These two figures are not meant to be complete but to illustrate the complexities involved in creating an analysis model such as those used to determine solder joint strain.

Not all dimensions are independent, and it is acknowledged that making a good combination of modeling decisions for a given problem is non-trivial.

- Enclosure
 - cooling method
 - PWA location, support structure
- PWA
 - component locations/density
- PWB
 - thickness: $0.020 < t < 0.125$
 - materials: FR4, copper
 - # of layers, layer orientation
- Component Occurrence
 - location, side of board
 - soldering process: wave soldering (epoxy dot), reflow soldering
- Component
 - function: resistor, capacitor, microprocessor
 - materials: alumina, plastic
 - body style:
 - surface mount
 - leadless: discrete chip (0805, 1206), LCCC (20, 52, 156 pin), MELF
 - leaded: J-lead, gull wing, SOT, flat pack
 - through-hole: axial, radial, DIP

Figure 9.5 Example PWA Product Variations

Given such variations, one can appreciate that there are several possible "good" analysis models depending on the purpose of the analysis and the product values involved. One model may require solving simple formulas while another may involve a complex finite element analysis solution. The former may be appropriate for early design comparisons while the latter may be better suited for detailed analysis later in the design process.

If the Coffin-Manson model is used in Step 1, then the goal of this step is to determine the strain range the solder joint experiences each load cycle, $\Delta\epsilon^P$. Therefore,

some measure of cyclic strain must be extract from the thermomechanical state found in the analysis model. Figure 9.6 illustrates representative analysis models from the literature that vary in both regional resolution and complexity level (repeated here from Chapter 4). Models designated Levels 1-4 could all be used to determine the thermomechanical state of the component occurrence. The two models used in the case studies (Levels 1 and 3) that fulfill this step are described next.

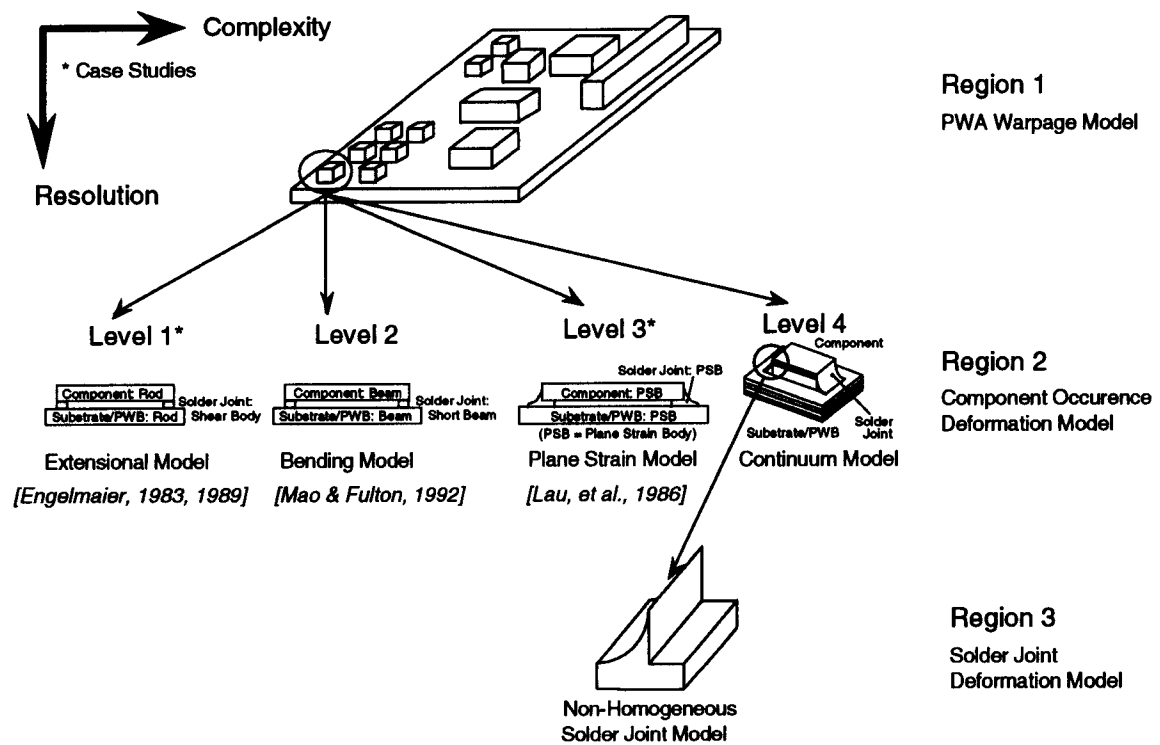


Figure 9.6 Varying Levels of Thermomechanical Analysis Models

Level 1: Extensional Model [after Engelmaier, 1983, 1989]

Engelmaier developed the following relations by assuming the solder joint undergoes uniform shear strain (Figure 9.7). Due to the viscoplastic behavior of the solder, at steady state the component and PWA are assumed to expand fully and unhindered as simple rods. In this case the more complicated viscoplastic material model actually simplifies the

analysis and is not needed explicitly. It is assumed that the primary materials dominate the behavior of the component and PWB (modeled as simple homogenous bodies). These dominate materials (alumina for ceramic components and FR4 for PWBs) are assumed to be linear elastic. The CTEs needed in these relations are given in Table 9.1.

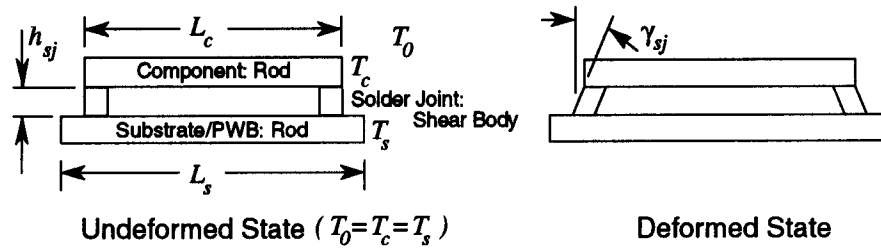


Figure 9.7 Level 1 Extensional Model
[after Engelmaier, 1983, 1989]

$$\gamma_{sj} = \frac{L_c \Delta(\alpha \Delta T)}{2h_{sj}} \quad (9.4)$$

$$\Delta(\alpha \Delta T) = \alpha_s (T_s - T_0) - \alpha_c (T_c - T_0) \quad (9.5)$$

Table 9.1 Case Study Material Properties
[Engelmaier, 1983; Lau, et al., 1986]

	E (psi)	ν	α (in/in-°C)
Alumina	37.0e6	0.30	6.7E-6
FR4	1.6e6	0.28	15E-6
Solder	1.5e6	0.40	21E-6
$\sigma_Y = 5000$ psi, $\lambda = 0.1$			

The following relations slightly generalize Engelmaier's measure of worst-case distance between solder joints. Of course more precise relations could be developed based on detailed component geometry, but it is not clear if the model itself warrants such accuracy.

$$L_c = L_{total} \quad \text{a. discrete surface mount components} \quad (9.6a)$$

$$L_c = \sqrt{L_{total}^2 + w_{total}^2} \quad \text{b. rectangular surface mount chip carriers} \quad (9.6b)$$

Engelmaier bases the solder joint height on the following heuristic, where $t_{solder\ stencil}$ is the thickness of the solder stencil used to screen solder paste onto the PWB.

$$h_{sj} = 1/2 t_{solder\ stencil} \quad (9.7)$$

Thus, the solder joint height could be linked to the detailed design model of the actual solder stencil. This relation demonstrates the need for manufacturing process information to support analysis during design. Another solder joint height heuristic for wave soldered components needs to be determined. Note, however, that this analysis model does not consider the effects of the epoxy dot that typically secures a component if it is wave soldered (similarly, conformal coating is not considered). The present model most likely would overestimate the solder joint shear strain in a wave soldered component because of increased stiffness; therefore, another model may be needed anyway in this case.

Since it is assumed that strain is uniform in the solder joint and that plastic deformation dominates, the strain range needed by Step 1 is given by the following two equations:

$$\Delta\gamma_{sj} = F |\gamma_{sj}| \quad (9.8)$$

$$\Delta\epsilon^p = \Delta\gamma_{sj} \quad (9.9)$$

where F is a correction factor based on experimental results. This factor depends on the type of solder joint per the following table (which is itself a discrete relation).

Table 9.2 Strain Range Correction Factor

Solder Joint Type	F [Engelmaier, 1989]
SMD chip	0.7 - 1.2
castellated leadless	0.7 - 1.2
columnar leadless	1.0 - 1.5
leaded	1.0

Level 3: Plane Strain Model [after Lau, et al., 1986]

The plane strain model (Figure 9.8) was developed by Lau, et al. to study the effects of interconnection geometry on the solder joint fatigue of a surface mount chip resistor mounted on an FR4 PWB. Since more geometric detail is considered, a finite element-based solution is required. Solder was modeled as a bilinear kinematic hardening material [SASI, 1990] with properties given in Table 9.1. Figure 9.9 illustrates the parameters they used to model the solder joint geometry.

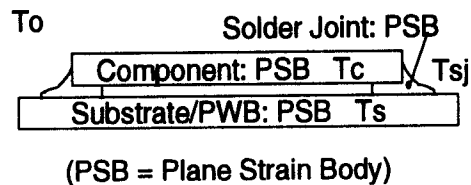


Figure 9.8 Level 3 Plane Strain Model
[after Lau, et al., 1986]

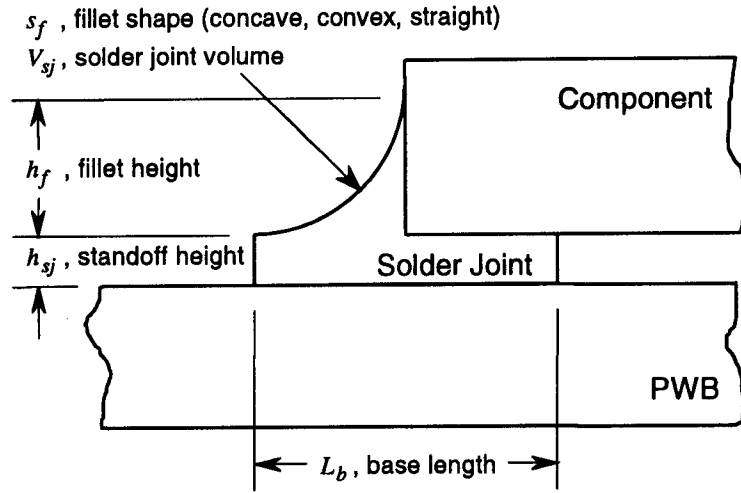


Figure 9.9 Idealized Solder Joint Geometry

In the present research the following extensions were made to the plane strain model to supplement Lau, et al.'s model. The primary intent of these extensions was to illustrate the capabilities of the PBAM representation (i.e., no claim is made regarding how good these extensions are with respect to the determination of solder joint fatigue itself).

- a. To demonstrate how PBAMs can support different types of analysis modeling options, alternative models with linear elastic solder behavior and/or rectangular solder joint geometry were allowed.
- b. Surface mount leadless components beyond just chip resistors are allowed. Therefore, Eqn. 9.6 is used to determine L_c .
- c. To support different component lengths, the following simple relation was added.

$$L_s = 1.25L_c \quad (9.10)$$

- d. When the nonlinear solder option is chosen, the extensional model is used to estimate the initial load step in the finite element analysis. The load yield factor, a , is the factor by which the thermal load would be scaled to cause the solder joint stress to equal the

yield stress. The details of the relation that determines a are given in Appendix F under the Interconnected Rods System. Other relations can be found in the *Ansys Theoretical Manual* (e.g., number of load steps).

As in the extensional model, the component and PWB are considered to be homogenous bodies consisting of their dominate material which are again modeled with isotropic linear elastic stress-strain behavior (Table 9.1). Though no closed solution is known to exist for this analysis model and associated variations, it is still helpful to acknowledge the existence of the following relations. In fact, the input/output tuples obtained by running multiple FEA analyses with different values would be discrete relations in the truest mathematical sense [Bender and Williamson, 1991].

$$r_1 (T_o, L_c, h_c, E_c, \nu_c, \alpha_c, T_c, L_s, h_s, E_s, \nu_s, \alpha_s, T_s, L_b, h_{sj}, E_{sj}, \nu_{sj}, \alpha_{sj}, T_{sj}, \gamma_{xy \text{ extreme, } sj}, d) \quad (\text{rectangular s.j., linear solder}) \quad (9.11)$$

$$r_2 (T_o, L_c, h_c, E_c, \nu_c, \alpha_c, T_c, L_s, h_s, E_s, \nu_s, \alpha_s, T_s, L_b, h_{sj}, h_f, V_{sj}, s_f, E_{sj}, \nu_{sj}, \alpha_{sj}, \sigma_{Y, sj}, \lambda_{sj}, T_{sj}, \gamma_{xy \text{ extreme, } sj}, d, n, a, e) \quad (\text{detailed s.j., nonlinear solder}) \quad (9.12)$$

Eqn. 9.11 is for the case of rectangular solder joint geometry and linear elastic solder, while Eqn. 9.12 is for detailed geometry (Figure 9.9) and bilinear kinematic hardening solder. Similar relations could be written for the other combinations (detailed s.j., linear solder; rectangular s.j. & nonlinear solder). Note the inclusion of solution method parameters d, n, a, e since they affect the analysis results (d is a measure of mesh density, n is the number of load steps, and e is the convergence criteria).

Though other variables could be included in the above relations (e.g., fields of deformation, stress, and strain) it is the extreme total shear strain in the solder joint, $\gamma_{xy \text{ extreme, } sj}$, that is of interest .

$$\Delta \gamma_{sj} = \left| \gamma_{xy \text{ extreme, } sj} \right| \quad (9.13)$$

Though not explicitly stated, Lau and co-workers apparently adopted the above relation and Eqn. 9.9 (to provide input into the Coffin-Manson relation) by assuming that all strain becomes plastic strain at steady state.

The following list summarizes the information required by the plane strain model beyond that needed by the extensional model. The number of additional relations and variables is one measure of relative model complexity, along with what type of solution methods the relations require.

- Component and PWB geometry: h_c, L_s, h_s
- Solder joint shape: $L_b, h_{sj}, h_f, V_{sj}, s_f$
- Material properties: $E_c, \nu_c, E_s, \nu_s, E_{sj}, \nu_{sj}, \alpha_{sj}, \sigma_{Y,sj}, \lambda_{sj}$
- Initial load step estimator
- Solution method parameters: d, n, a, e

4. Determine Global PWA Thermomechanical Loads

This step would consider the interaction of components, solder joints, PWB board layers, and conductive traces that could cause the PWA to warp. The basic idea is to get warpage (out-of-plane deformation) and in-plane deformations from this global warpage model around the component of interest. These values then would be used as boundary condition inputs to the local model of Step 3 (Figure 9.2).

Yeh, et al. [1993] and Garratt [1993] have shown that the copper traces on a simplified bare PWB contribute significantly to global PWB warpage. However, for a realistic PWA, including the numerous conductive traces in a finite element model (along with component and solder joint details) most likely would make the model too large for solution. No known analysis model currently exists which considers such effects. Solomon [in Lau, 1991, p. 438] refers to work by others that determined the magnitude of

PWA bending likely to occur. He notes that bending is negligible at high temperatures or low strain rates, but that it becomes more prominent in the opposite cases.

To test how global/local models could be represented as PBAMs, an analysis model for PWA warpage was developed in this research at a high-level information input/output level. Only the PWA design information that would be needed along with the information interfaces between this global model and the local model (Step 3) were considered. Hence, this thesis contains numeric results only for the case where warpage effects are neglected.

5. Determine Global PWA Thermal State

For the case study analysis models, the goal of this step is to determine the spatially averaged component and PWB temperatures under the given thermal loading conditions. If spatial temperature distributions are desired (e.g., for input into a more detailed strain model in Step 3) another possible step would be determining the local component occurrence thermal state.

Two basic types of thermal loads are considered that are relevant to solder joint fatigue [Engelmaier, 1989, 1983]:

A. *Uniform Thermal Cycling* This load can result from daily temperature cycles experienced by products in non-climate-controlled environments, such as those which occur outside or in a warehouse. No analysis model is needed if steps 2, 3, and 4 only require steady state conditions since the following equation will hold at steady state.

$$T_c = T_s = T_{ss} \quad (9.14)$$

B. *Power Cycling* Turning on and off a personal computer everyday is perhaps the most familiar example of this type of load. Before the product is turned on, the whole PWA is typically at a uniform temperature, T_o . After it is turned on, a temperature

difference between the component and the PWB will typically exist, causing strain in the solder joint even if the CTEs are perfectly matched [Engelmaier, 1983]. Engelmaier's steady state thermal expansion mismatch, $\Delta(\alpha\Delta T)$, is a better measure of thermomechanical difference between the substrate and component than just CTE difference because it includes the temperatures of each body.

In this case (and the thermal cycling case if transient loads are to be modeled), information about the electrical circuitry; the thermal properties of the components, PWB, and component interfaces (solder joint, adhesive, heat sinks, etc.); and the enclosure thermal environment are needed to define the thermal analysis model. Collectively this information would be contained in the PWA occurrence, ω_{pwa} .

Again, as in Step 3, this analysis can involve simple formulas or complex models requiring discretization for solution. The following conceptual relation is part of this model, which typically would be solved approximately using a tool such as Autotherm [MGC, 1991]).

$$r(T_o, \omega_{pwa}, \omega_c, T_c, T_s) \quad (9.15)$$

As the relation between component junction temperature and component reliability has been known for quite some time, computer-aided tools that determine the thermal state of a PWA are well developed. Tools for this purpose exist that are specially integrated into the PWA design process (e.g., Mentor Graphics Autotherm [MGC 1991b]). Therefore, Engelmaier gives minimal details on determining the following power cycling temperatures used in his ceramic chip carrier (CCC) example: $T_c = 96^\circ\text{C}$, $T_s = 88^\circ\text{C}$.

The plane strain model by Lau, et al. only considered the steady state thermal cycling case (from -55°C to 125°C , as in automobile under-the-hood conditions [Engelmaier, 1989]), so their model required no thermal analysis. In this thesis, power

cycling was also applied to the plane strain model by estimating the uniform solder joint temperature as follows:

$$T_{sj} = \frac{1}{2}(T_c + T_s) \quad (9.16)$$

When the component being analyzed is a chip resistor, this thesis estimated the corresponding power cycling temperatures to be approximately $T_c = 90^\circ\text{C}$ and $T_s = 88^\circ\text{C}$. The reference temperature used is $T_o = 20^\circ\text{C}$.

9.3 Development of PBAMs for Analysis of Solder Joint Fatigue

This section shows how multiple PBAMs were developed to represent the case study analysis models by Engelmaier and Lau, et al., described in the previous section. As discussed in Chapter 6 (Example 6.1), a product model either must exist or must be created in the process of developing PBAMs. A preliminary PWA product model is described in Appendix E that has been used in these case studies, as well as in other PWA design-related activities such as selection of electronic components [Peak and Fulton, 1992a]. The reader is encouraged to review Appendix E at this point to enable better understanding of this section.

Figure 9.10 is an object relationship diagram (using EXPRESS-G) showing the relationships between these PBAMs and the analytical building blocks they utilize. This view is derivable from the populated PBAM structure of each PBAM. These PBAMs correspond with the analysis steps in Figure 9.1 as highlighted here in reverse step order (bottom-up).

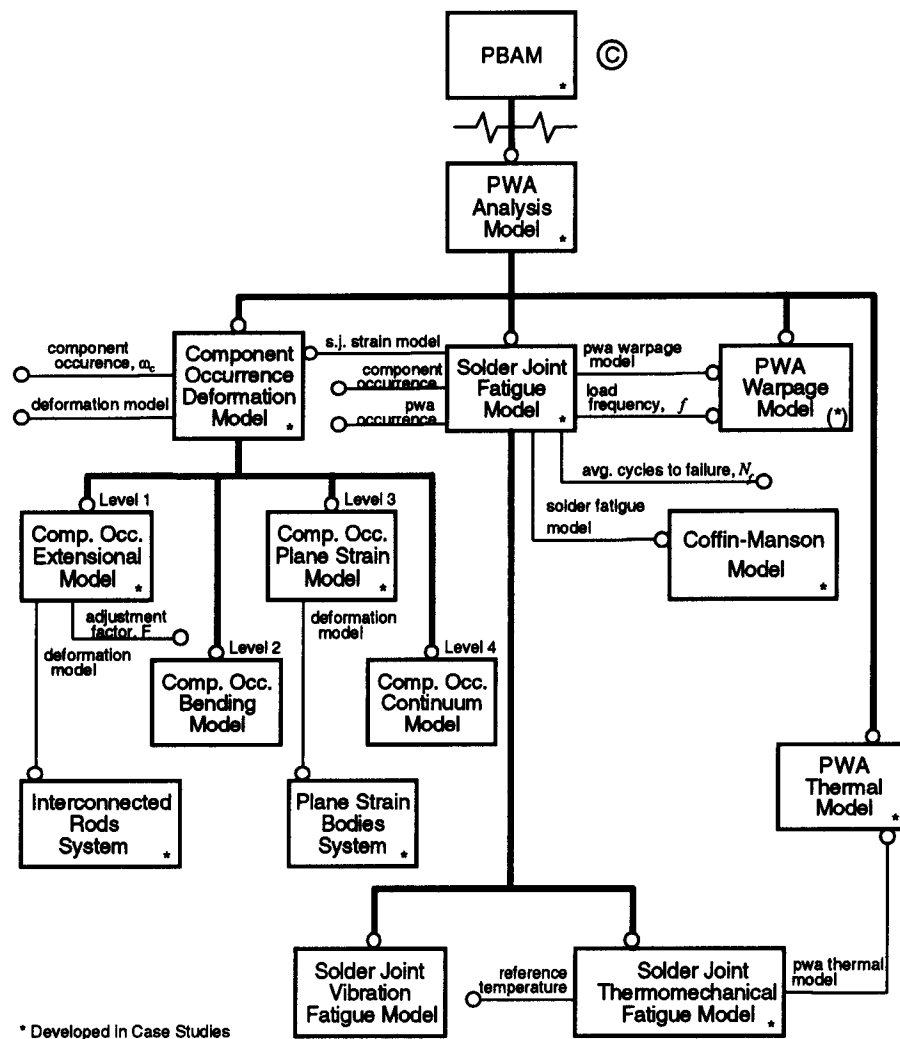


Figure 9.10 PWA Analysis Model Object Relationship Diagram

STEP 5 The PWA Thermal Model is a PBAM that provides component and substrate (PWB) temperatures when a PWA is under operational (i.e., powered) loads.

STEP 4 The PWA Warpage Model PBAM would provide global warpage values into the local deformation model of Step 3. It was developed as an extension at a conceptual level only and will be discussed later in this Chapter.

STEP 3 The Component Occurrence Deformation Model is an *abstract* [ISO 10303-11] PBAM. Abstract means that one of its subclasses can be instantiated for use, but it cannot itself (see Appendix A). Figure 9.6 shows four models of varying complexity level that determine deformation in a component-solder joint-PWB assembly (a component occurrence). Per Figure 9.10 all four models are subclasses of this abstract class (Figure 9.10) which was developed to capture the information these deformation models have in common. In this thesis only the Level 1 and 3 PBAMs have been developed as case studies. Figure 9.17 illustrates differences and samenesses between these two PBAMs.

LEVEL 1 The Component Occurrence Extensional Model (a. k. a. the Extensional Model) is a PBAM that represents component occurrence deformation behavior, where the component and PWB are modeled as rods. Thus, this PBAM includes the relations in Engelmaier's model that determine solder joint strain under thermomechanical loads (as well as other relations).

LEVEL 3 Component Occurrence Plane Strain Model (a. k. a. the Plane Strain Model). This PBAM performs the same function as the preceding PBAM, except all parts are modeled as bodies with plane strain behavior. Since this PBAM also allows different solder stress-strain behaviors and varying solder joint geometry detail, it represents a generalized version of the strain model by Lau, et al.

Step 1 below discusses how a solder joint fatigue PBAM uses these two PBAMs.

STEP 2 Solder property determination is described next.

STEP 1 The Solder Joint Thermomechanical Fatigue Model (SJTF Model) is a special type of the Solder Joint Fatigue Model. It wraps and connects the above PBAMs to predict solder joint life under thermomechanical loads. It takes the temperatures from the thermal PBAM and processes them for input into the Coffin-Manson Model which

determines the solder properties (STEP 2). It also links the strain determined by either of the above deformation PBAMs into the Coffin-Manson fatigue relation. Finally, the fatigue life can be output if the load frequency and component occurrence are input (with respect to a design verification input/output viewpoint).

One challenge of representing the analysis models is determining where to put each relation and the data it utilizes. Generally, one should balance complexity against grouping relations that are associated with each other. Relations that are likely to be used repeatedly as a group can be broken out from an otherwise associated larger group. One should also place relations at the correct level of generality. Finally, one must keep in mind that some of the information used by the PBAMs also is needed by other design and analysis tasks (e.g., component selection [Peak and Fulton, 1992a]). Therefore, the proper representation of this information to support such heterogeneous utilization is important to a flexible and extendible design environment.

Examples of how such guidelines were applied in the case studies is included in the following descriptions of the three major PBAMs. The reasoning behind the placement of each relation is also given next. Supporting analytical primitives and systems used for the case studies are marked with an asterisk (*) in the figures of Appendix F. Appendix G contains the views of the above PBAMs.

Solder Joint Thermomechanical Fatigue Model (SJTF Model)

This PBAM is capable of determining leadless component solder joint fatigue life under power cycling and elevated thermal cycling. The Solder Joint Fatigue Model superclass contains information that would also be common to the Solder Joint Vibration Fatigue Model shown in Figure 9.10.

Figure 9.11 shows the SJTF Model in an instance view (defined in Chapter 5 and 6) which has been annotated to show where a few example equations are represented. A bold border surrounds the constraint schematic of this PBAM, which is an example of a complex PBAM. The analysis context specifies the PBAM options (described below). The values in the analysis context show how a specific example product entity (R110) and analysis entities (frequency, reference temperature, and steady state temperature) can be connected to the PBAM as inputs (indicated by the arrow directions). After that, the fatigue life, \bar{N}_f , is automatically determined as an output. This figure shows sample product and analysis entities for the case of thermal cycling and Extensional Model usage (Case #1 - see Table 9.3).

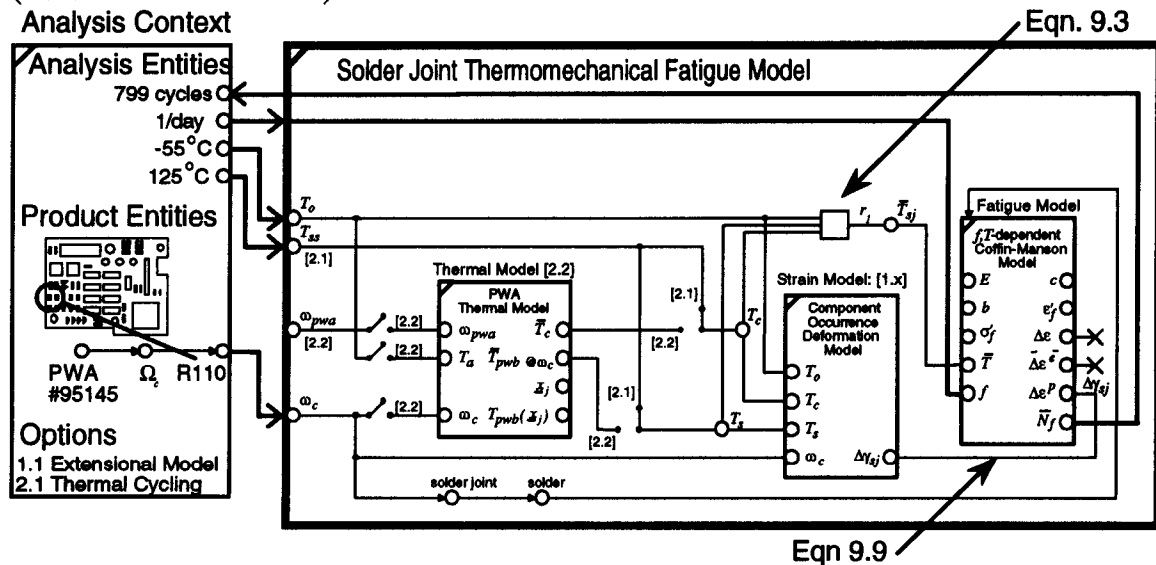


Figure 9.11 Solder Joint Thermomechanical Fatigue Model Instance View

The major steps required in a solder joint fatigue analysis are represented by the three subsystems shown: Thermal Model (Step 5), Strain Model (Step 3), Fatigue Model (Steps 1 & 2). Figure 9.12 shows one possible subsystem view of the SJTF Model itself which could be used potentially by another PBAM. These subsystems and connections will be described now from the point of view of Figure 9.11 where the fatigue life is the output.

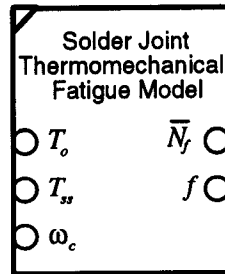


Figure 9.12 Solder Joint Thermomechanical Fatigue Model Subsystem View

The analysis context specifies the desired type of thermal load as an option in the **SJTF Model** (option category 2 in the Figure 9.11). In the case of thermal cycling, the "switches" are in the [2.1] position as shown, and the subsystem labeled **Thermal Model** is bypassed since Eqn. 9.14 applies. If the power cycling option were selected, the "switches" would be in the [2.2] position, and the **Thermal Model** would determine the component and PWB temperatures via Eqn. 9.15.

In either case the **SJTF Model** connects these temperatures directly to the **Strain Model**. Since Eqn. 9.3 is specific to the **SJTF Model**, it is represented as relation r_I therein and transforms the temperatures for input to the **Fatigue Model**.

This PBAM could be considered a generalized version of Engelmaier's full fatigue model since the subsystem labeled **Strain Model** can be the **Extensional Model** (as in his model) or the more complex **Plane Strain Model**. Originally the **Extensional Model** and the **SJTF Model** were one PBAM (the **PWA Two Rod Model** in [Peak and Fulton, 1992b]). As determining the strain in a solder joint is a relatively major step in the overall fatigue analysis process, that original PBAM was split into the two current PBAMs. This split became even more advantageous when the **Plane Strain Model** was added; otherwise, another PBAM, e.g., a **PWA Plane Strain Model**, would have been needed. Instead, the current approach was adopted to limit the complexity contained in any one PBAM and to

increase modularity. Thus, the SJTF Model has an option category to specify the deformation model used. This situation is an example of **subsystem substitution** (Chapter 5) and is indicated by the [1.x] label in the constraint schematic. As indicated in the analysis context, the Extensional Model (option 1.1) is used in this particular instance view.

The Strain Model determines the solder joint strain range, $\Delta\gamma_{sj}$. Note that the SJTF Model connects this variable to the Fatigue Model by representing Eqn. 9.9 as a simple equality relation (a solid line). The Fatigue Model then uses the frequency, f , to finally determine the fatigue life, \bar{N}_f . In Figure 9.11, sample product and analysis entities are shown for the case of thermal cycling. Note that product variables and analytical variables would be supplied to all three subsystems in this PBAM (if the power cycling option were chose), but this need not be true for all complex PBAMs.

Eqn. 9.1 is captured as a relation in the Coffin-Manson Model class which can be used for applications other than just solder joint fatigue. A Coffin-Manson Model can be associated with all materials for which it is applicable. Since Eqn. 9.2 and the value for ϵ_f' are specific to 60%Sn-40%Pb and eutectic solder, they are stored in representations of those solders. Another PBAM could be developed to wrap the generic Coffin-Manson Model, just as the Level 1 and 3 models wrap their generic analytical systems; however, the small number of connections to the Fatigue Model did not seem to warrant an extra PBAM.

Finally, note the extended constraint graph view of the SJTF Model given in Figure 9.13. Dashed lines encircle the subsystems, and applicable options are indicated. This view can be used to trace what outputs can be obtained given certain inputs to create I/O tables like the one given later in this chapter.

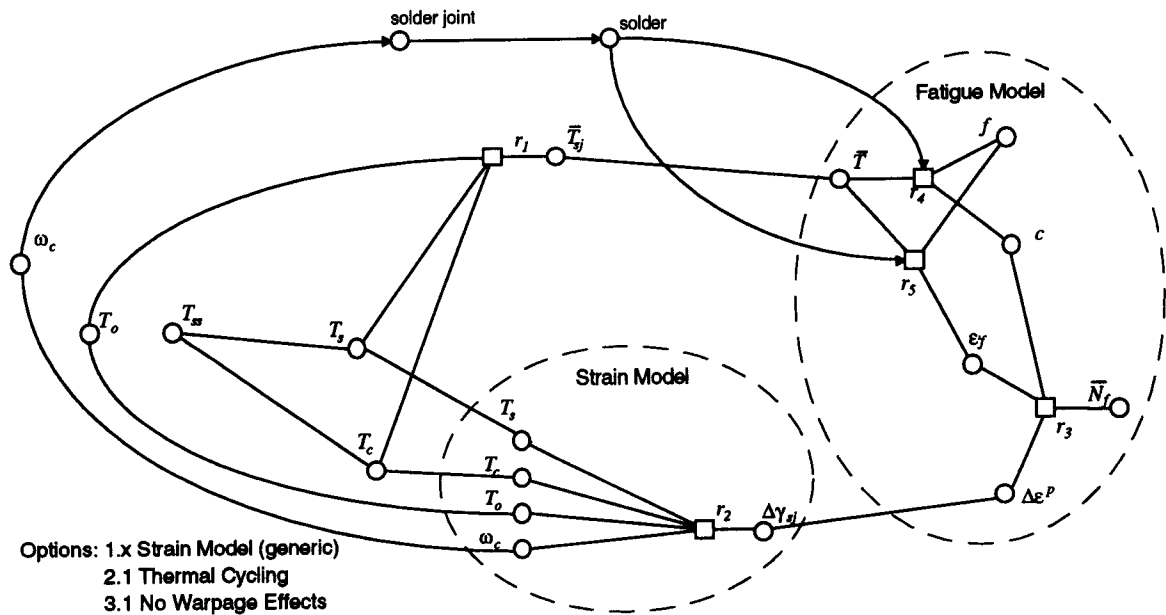


Figure 9.13 Extended Constraint Graph of SJTF Model under Thermal Cycling Load

Component Occurrence Extensional Model (Level 1)

Figure 9.14 gives an instance view of this simple PBAM. This PBAM uses the general purpose ABB Interconnected Rods System given earlier (see Example 5.3) as its Deformation Model subsystem. Basically the PBAM connects product variables in a component occurrence (e.g., R110) to the analytical variables in the generic Interconnected Rods System. For example, Eqn. 9.6 is a PAT represented in the constraint schematic as labeled in Figure 9.14. Eqn. 9.7 is represented similarly as indicated. The component occurrence is asked for the solder stencil thickness since it would know the manufacturing process from which to request the desired information; however, since such manufacturing objects are not supported in the current implementation, the solder stencil thickness is a variable in the Component Occurrence class.

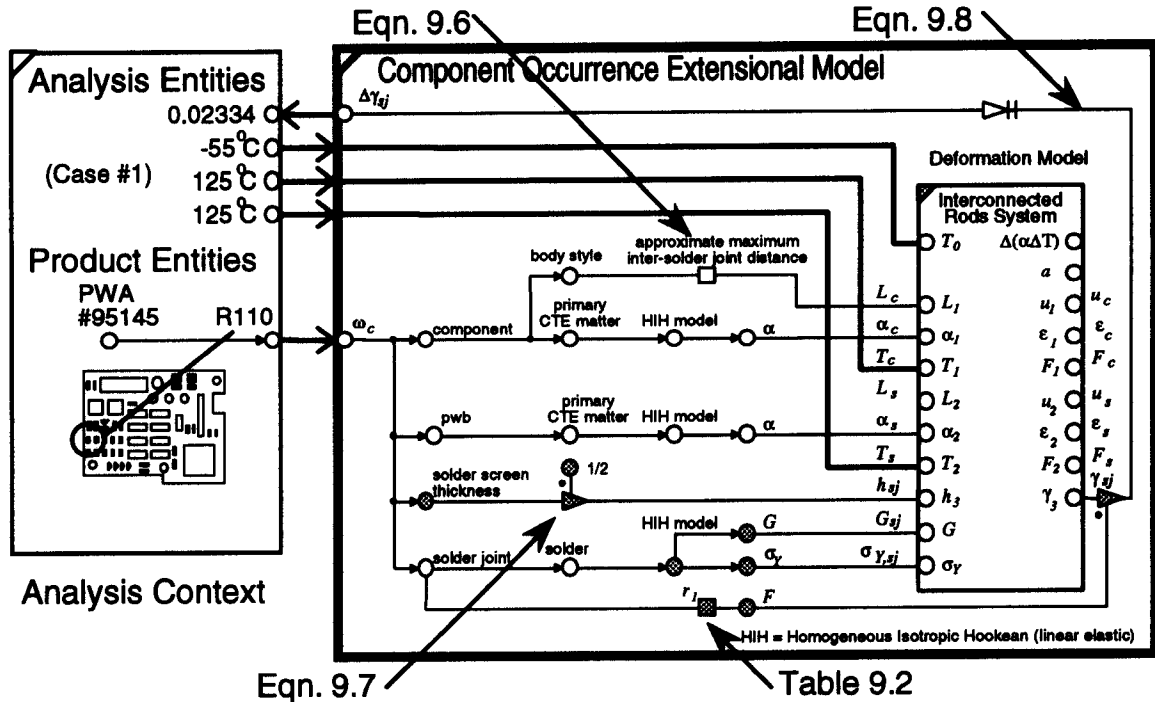


Figure 9.14 Extensional Model Instance View

The Interconnected Rods System contains generic deformation relations in its constraint schematic (see Eqns. 5.4 & 5.5 and figures in Example 5.3). The PBAM performs the semantic mappings from the application-specific relations (Eqns. 9.4 & 9.5) into these generic relations. For example α_s is mapped to α_l . Since analytical systems are "generic" components that can be used by many different PBAMs, this PBAM uses only some of the capabilities contained in the Interconnected Rods System. The material properties come from the component occurrence via product-analysis transformations contained in the Extensional Model.

Eqn. 9.8 adapts changes the shear strain into shear strain range as represented by the scale & offset relation and the absolute value relation shown in series. This equation is an example of an analysis-analysis transformation in the Extensional Model. Since the adjustment factor, F , is experimentally determined specifically for this model (Table 9.2), it is contained in the scope of this PBAM.

A subsystem view of this PBAM is given in Figure 9.15 which shows how semantically linked subsystem analytical variables can be included in this abstracted view for input/output connections if desired (depending on how one will use the PBAM.)

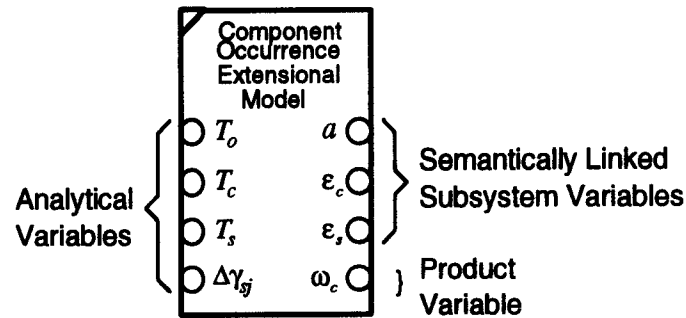


Figure 9.15 Extensional Model Subsystem View

Component Occurrence Plane Strain Model (Level 3)

The constraint schematic for this PBAM is given in Figure 9.16. Its subsystem, a Plane Strain Bodies System, is analogous to the Interconnected Rods System in the Extensional Model. It is this subsystem that contains the relations requiring FEA solutions (Eqns. 9.11 & 9.12).

Solder joint geometry variation is supported as option category 1 in the figure (as indicated by the switches in the figure), while category 2 is for the solder stress-strain model option. Note also the use of the Extensional Model as the Load Step Estimator subsystem when the nonlinear solder model option, [3.2] is chosen. Other relations are represented in a manner similar to that used in the Extensional Model.

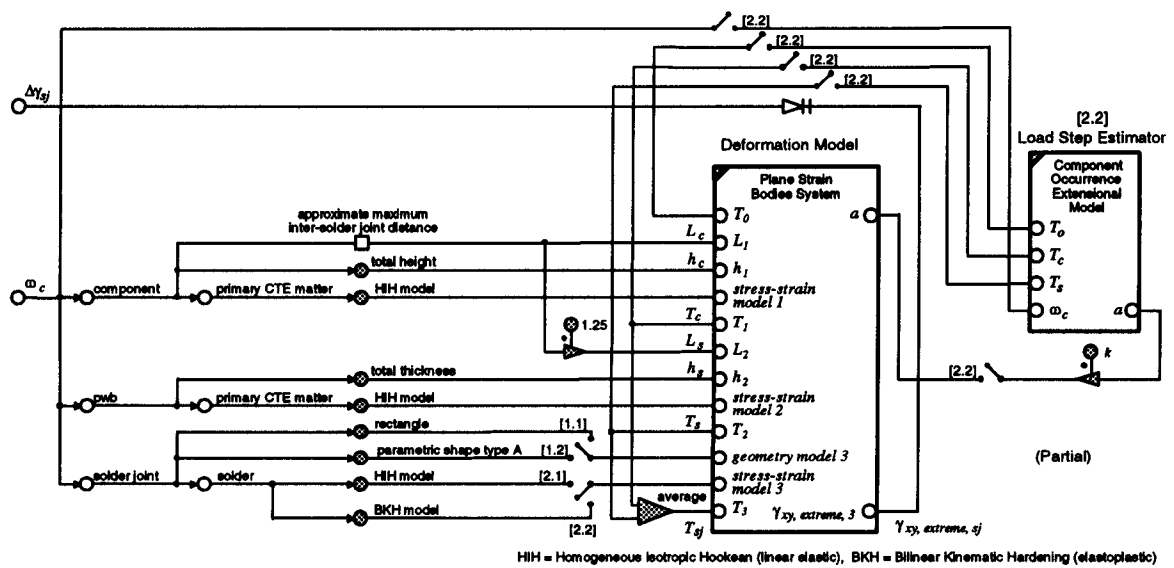


Figure 9.16 Plane Strain Model Constraint Schematic

Analysis Model Commonality

Another feature of PBAMs related to subsystem substitution is the exploitation of analysis model commonality. Commonality occurs frequently in analysis models that are at the same regional resolution for the same problem but that vary in complexity at that resolution. For example, the Level 1, 2, 3, and 4 models shown in Figure 9.6 have this characteristic. The more complex models are typically made by generalizing the types of loads, behaviors, geometry, and configurations that are allowed in the model. Thus, the simpler models may be special cases of the more complex models.

Figure 9.17 compares the Extensional Model with the Plane Strain Model. Common features are shown by shaded symbols, while unshaded symbols denote features that are new to that particular PBAM. Figure 9.10 shows the superclass PBAM that the two PBAMs in Figure 9.17 have in common to avoid redundancy and inconsistency errors.

9.4 Implementation of Case Study PBAMs

This section shows how the PBAMs developed in the previous section were implemented (see Chapter 8) in a prototype CAD/E framework which is described in Appendix D. First, the following implementation limitations should be noted (along with general framework limits given in Appendix D).

1. The current implementation does not exactly match the PBAM views given in Appendix F and G. Primarily, only the Extensional Model (per the note below) has been implemented using constraints, while the Plane Strain Model remains in an earlier (pre-constraint concept) single I/O alternative form (i.e., it only goes in the direction of determining fatigue life as the output). Options can be set procedurally in the Plane Strain Model. After implementing some of the key concepts, the specific case study PBAMs (as well as the definitions of the general PBAM structure, operations, and views) were further refined.

Also, the common superclass PBAM for both models (Comp. Occ. Deformation Model) was not implemented, so neither was the Deformation Model subsystem. Basically, the PWA Two Rod Model in [Peak and Fulton, 1992b] that was mentioned above still exists as a merging of the SJTF Model and the Extensional Model. However, its implementation was changed to utilize constraints and it does support a Fatigue Model subsystem. Thus, it can carry out the multidirectional design scenario mentioned later in this chapter.

2. Nonlinear Cases #4 and #14 in Table 9.3 were run by manually supplying input to a parameterized ANSYS Prep7 file. The automatic creation and execution of this file from the Plane Strain Model would involve a procedure similar to that used in the linear cases.

3. A "black box" thermal model containing a few typical datasets {thermal condition, reference temperature, component temperature, PWB temperature} was used (i.e., the thermal analysis to obtain input component and PWB temperatures was not actually performed in this system). Representative temperatures from Engelmaier [1983,1989] were used. It was implemented as a black box only in that it simply supplies the temperatures used in the above papers.
4. The ANSYS results retrieval link parses the results file to extract only the stress and strain extrema in the solder joint. If desired, the full ANSYS results file could be loaded and stored as STEP FEA entities [ISO 10303-104] using previously demonstrated techniques [Yeh, et al., 1991; Yeh, 1992].
5. The DeltaBlue constraint solver algorithm used in the prototype cannot handle cycles. However, techniques were performed to break the cycles and make it possible to get multiple I/O alternatives for the Extensional Model case. This was done by using two variables for the reference temperature which were not directly connected by constraints (thus, the cycle was broken). Since these variable were always inputs, their equality could be maintained imperatively in the method for setting the value of the reference temperature.

2.5 Representative Design and Analysis Scenarios

With the above limits in mind, this section shows how these PBAMs work under representative design and analysis scenarios using representative PWA and analysis datasets.

Design Verification Scenario

Figure 9.18 illustrates the overall process from a software and hardware implementation point of view. The following describes the execution of each step to support a typical design verification scenario.

1. As illustrated in Chapter 2, a designer ideally would like to perform design verification checks as the design evolves. Here it is assumed that the components are being laid out on a PWA using a tool like BoardStation by Mentor Graphics. To check the solder joint reliability on this PWA, the designer selects a PBAM to use and specifies which components to check. The present PWA layout could be automatically

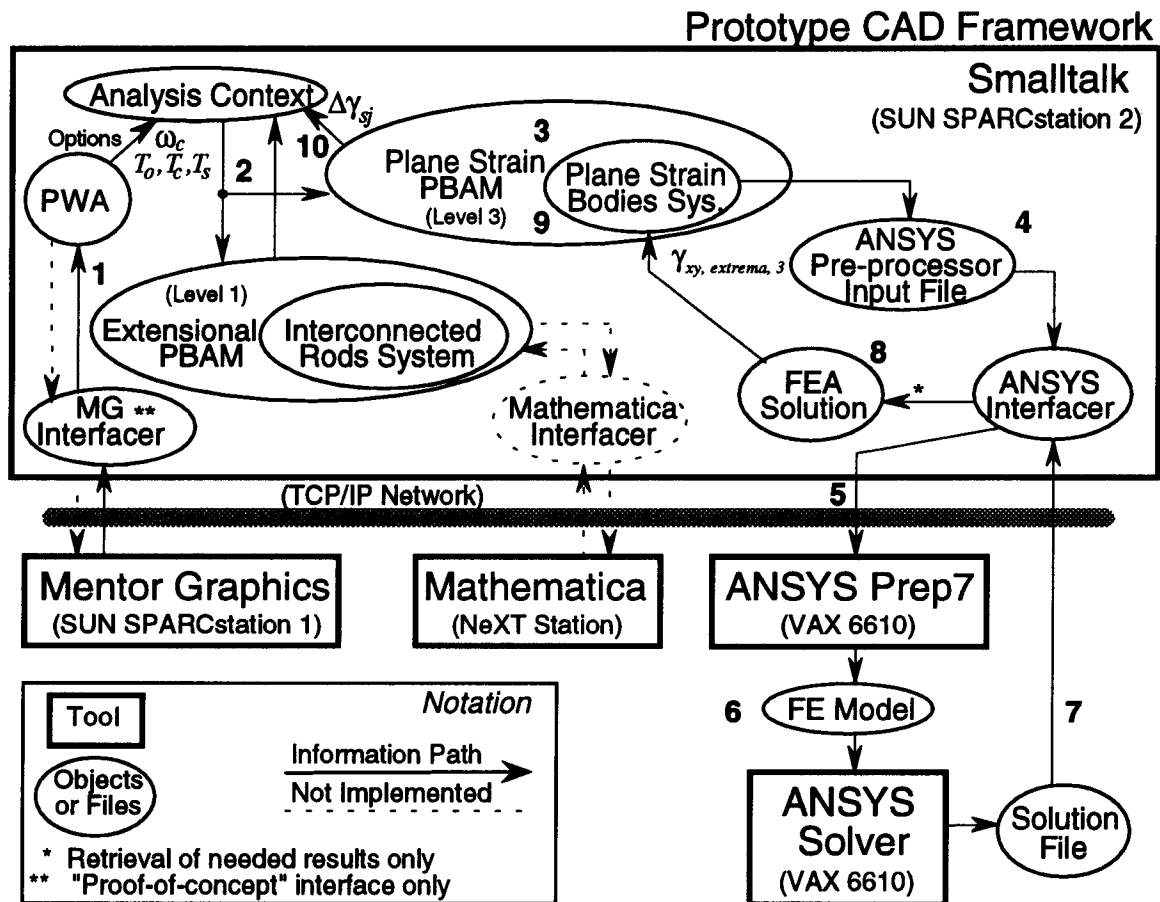


Figure 9.18 Prototype Implementation of Case Study PBAMs

transferred to the common database where the PWA object and related component objects have access to their other attributes that may not be used by Mentor Graphics (e.g., material properties and detailed geometry). The remaining steps are functional in the prototype.

2. To keep the figure from becoming even more cluttered, it is assumed that only the extreme solder joint shear strain range is to be determined (so only the Level 1 and 3 PBAMs for this purpose are shown). Effectively the designer is the analysis context in this scenario. He or she can specify the temperature conditions for the analysis and choose the component occurrence, ω_c , to be checked.

Assuming the Plane Strain Model PBAM is selected, options for geometric and material detail can be specified. Finally, the designer tells the PBAM which I/O combination to use to get the maximum shear strain as the output. No further user intervention is required except as noted below.

3. With the above inputs, the Level 3 PBAM is ready to go. The PBAM instantiates a Plane Strain Body System and supplies it with needed data extracted and transformed from the component occurrence and temperature inputs (Figure 9.16). Then the PBAM asks this subsystem for the maximum shear strain in its interface body (the subsystem does not know that the interface body is a solder joint - the PBAM keeps track of that).
4. The subsystem knows that it needs to get the requested answer via a finite element solution, so it creates an ANSYS PREP7 input file [SASI, 1990] by filling in the appropriate blanks in a parameterized template. It then passes the results to the ANSYS interfacier.
5. This interfacier in turn transfers the file to a remote VAX and tells ANSYS to process the file.

6. The mesh generation, solution, and final results processing are performed by the ANSYS PREP7, solver, and POST1 modules respectively. At the expense of increased processing time, these phases can be displayed via X Windows if desired.
7. When ANSYS is done, the ANSYS interfacier retrieves the results. (If one chooses the graphical display option, at this point the human user must click one button to let the interfacier know ANSYS is done. It is rather difficult to make VMS and UNIX get work together. In batch mode, no user intervention at all is required).
8. After reading in the file, the interfacier calls an ANSYS parser (created using T-gen [Graver, 1992]) to extract out the needed results (stress extrema in the linear solder case and total strain extrema in the nonlinear case).
9. The Plane Strain Bodies System gives the result requested (extreme strain in body 3) to the PBAM (after transforming the stress into strain, in the linear case).
10. Finally, the PBAM takes the absolute value of the result (Eqn. 9.13) since strain range was requested and gives the final result to the analysis context.

Thus, this PBAM implementation fully automates the creation, execution, and results feedback of a representative finite-element-based routine analysis model. The Level 1 Extensional Model is formula-based, so no external tool is needed to solve the individual relations in it (in contrast with relations like Eqns. 9.11 and 9.12, which require finite element-based solutions) In both cases in a constraint-based implementation, the constraint solver can handle the *interaction* of the relations. The earlier implementation of the Extensional Model without constraints [Peak and Fulton, 1992b] captures the relations in one-way methods. Conceivably one could forgo a constraint-based implementation; however, multidirectional interaction of many relations would become more complicated and inflexible (knowledge and control become intertwined).

Sample Design Verification Analysis Results

Table 9.3 summarizes results from tests run using representative datasets. All cases in this table were done from the design verification perspective described above where fatigue life is the desired output. Some variations (geometric and material transformations) within the Plane Strain Model (Level 3) are included in the table which illustrate PBAM flexibility. Also two types of thermal loads are supported (Thermal Cycling and Power Cycling), demonstrating the use of PBAM Options.

Figure 9.19 is a standard instance view of Case #1 in Table 9.3. Note how the arrows on the jumpers (bold connecting lines) indicate the inputs and outputs of the PBAM, and how the selected options are listed beside the PBAM. Values for inputs and the output (fatigue life) are indicated in the analysis context.

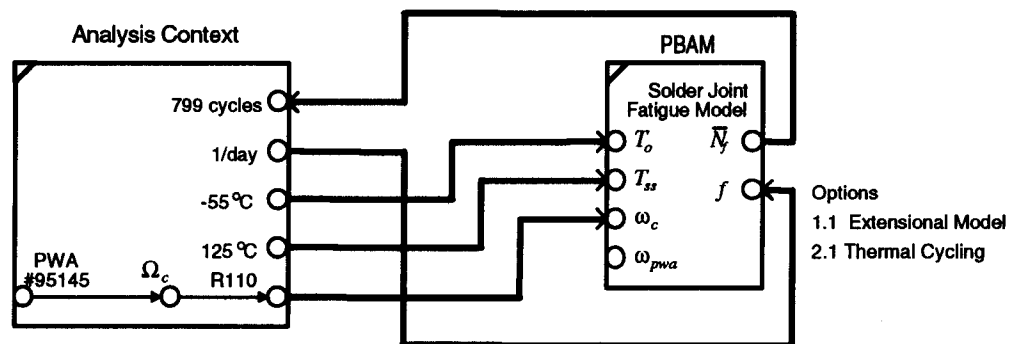


Figure 9.19 Standard Instance View of SJTF Model

Figures 9.20 through 9.22 illustrate results from various scenarios (as indicated by the Case #) within this Case Study. Figure 9.20 shows a view of the SJTF Model with Extensional Model applied to all the leadless components on a sample PWA. The results show that the leadless ceramic chip carrier (LCCC) poses a greater reliability problem than the resistors, as could be expected due to its greater length. Figure 9.21 is similar to Case

#62 where the rectangular solder joint geometry option was selected. The detailed solder joint geometry option was selected in Case #3 as shown in Figure 9.22 (which is a linear version of Figures 7 and 16 by Lau, et al. [1986]).

PWA Leadless Component Solder Joint Reliability Checker - Two Rod Model

PWA Listing

95235 PWA Test 2
95240 PWA Test 3
95255 PWA Test 4

Model Reference: Engelmaier, 1989
Reference Temperature (T_0 , degrees C): 20
Solder Joint Height (h, inches): 0.01
Adjustment Factor (F, unitless): 1.25
Power-up Cyclic Frequency (f, cycles/day): 1
PWB Temperature (T_s , degrees C): 88

Undeformed State


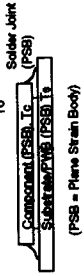
Deformed State

Ref. Des.	Part No.	Class	Body Style	T_c	Avg. Fatigue Cycles	Avg. Life (years)
R103	99200	Resistor	axial		*	*
R109	99120	Resistor	surfaceMount	89	7381.92	20.22
R110	99120	Resistor	surfaceMount	90	7553.77	20.69
R112	99120	Resistor	surfaceMount	90	7553.77	20.69
U102	91200	Microprocessor	LCCC	96	399.321	1.094

NEW CANCEL SAVE DELETE DONE

Figure 9.20 Solder Joint Reliability Checker Using PWA Two Rod Model

Table 9.3 Solder Joint Fatigue Case Study Analysis Results

SJTF Model Options	Level 1 Extensional Model 	Level 3 Plane Strain Model 
Solder Model	Viscoplastic Solder Model	Nonlinear Solder Model*
Solder Joint Geometry	1D Solder Joint	Detailed Solder Joint
Scenario	$\Delta\gamma_{sj}$ (strain) \bar{N}_f (cycles)	$\Delta\gamma_{sj}$ \bar{N}_f $\Delta\gamma_{sj}$ \bar{N}_f
Thermal Cycling ($h_{sj}=0.005"$, $0.062"$ FR4 PWB, $T=-55$ to $+125$ °C, $f=1$ cycle/day)		
1206 Resistor	#1 0.0233	#2 0.0270
LCCC-52	#51 0.1716	#52 0.0483
		#3 0.0098
		***#4 0.0119
		5422
		3536
Power Cycling ($h_{sj}=0.010"$, $0.062"$ FR4 PWB @ 88°C , $T_o=20^\circ\text{C}$, $f=1$ cycle/day)		
1206 Resistor, $T_c=89^\circ\text{C}$	#11 0.0043	#12 0.0088
LCCC-52, $T_c=96^\circ\text{C}$	**#61 0.0293	#62 0.0167
		#13 0.0036
		#14 0.0025
		36444
		77105

* Not integrated in prototype (manually created ANSYS file). LCCC = leadless ceramic chip carrier
Published results: ** $\Delta(\alpha\Delta T) = 511$ ppm (exact match) [Engelmaier, 1983]

*** $\Delta\gamma_{sj} = 0.0143$, $\bar{N}_f > 2000$ cycles (analysis), $\bar{N}_f > 885$ cycles (experiment)

[Figures 7 and 16 by Lau, et al. 1986]

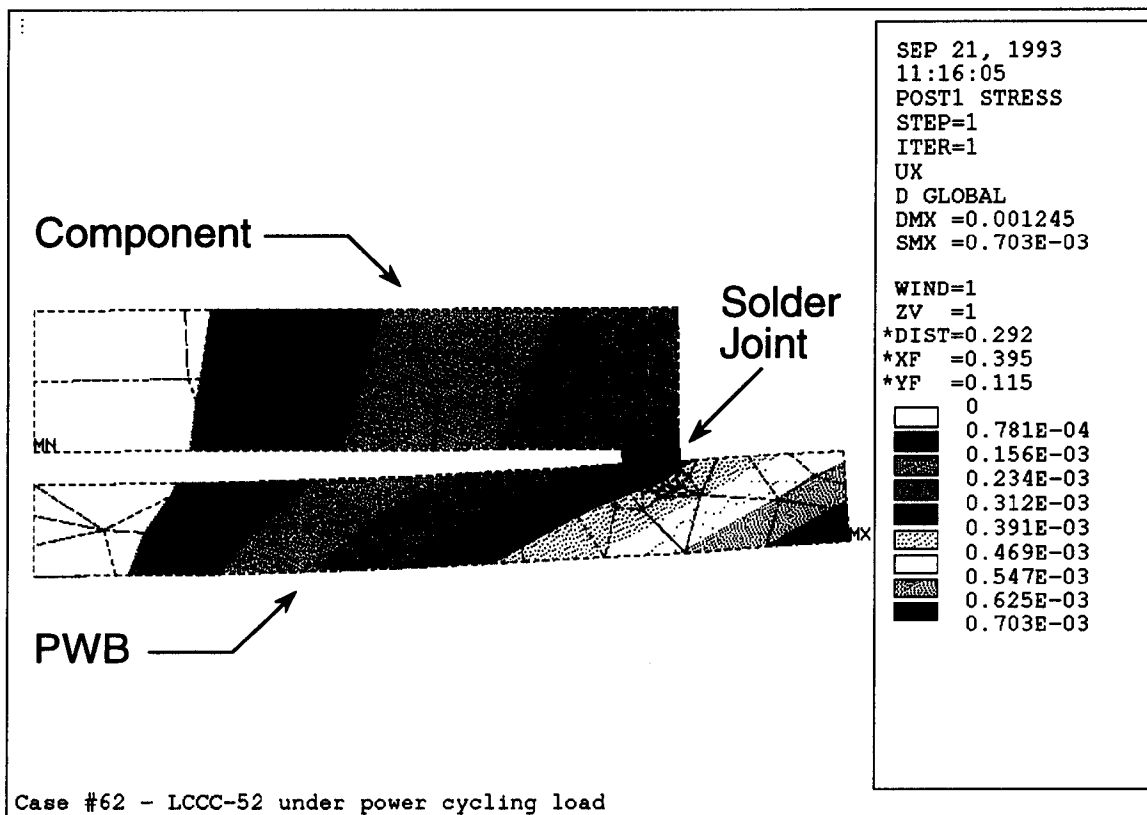


Figure 9.21 Deformations with Rectangular Solder Joint Geometry Option (Case #62)

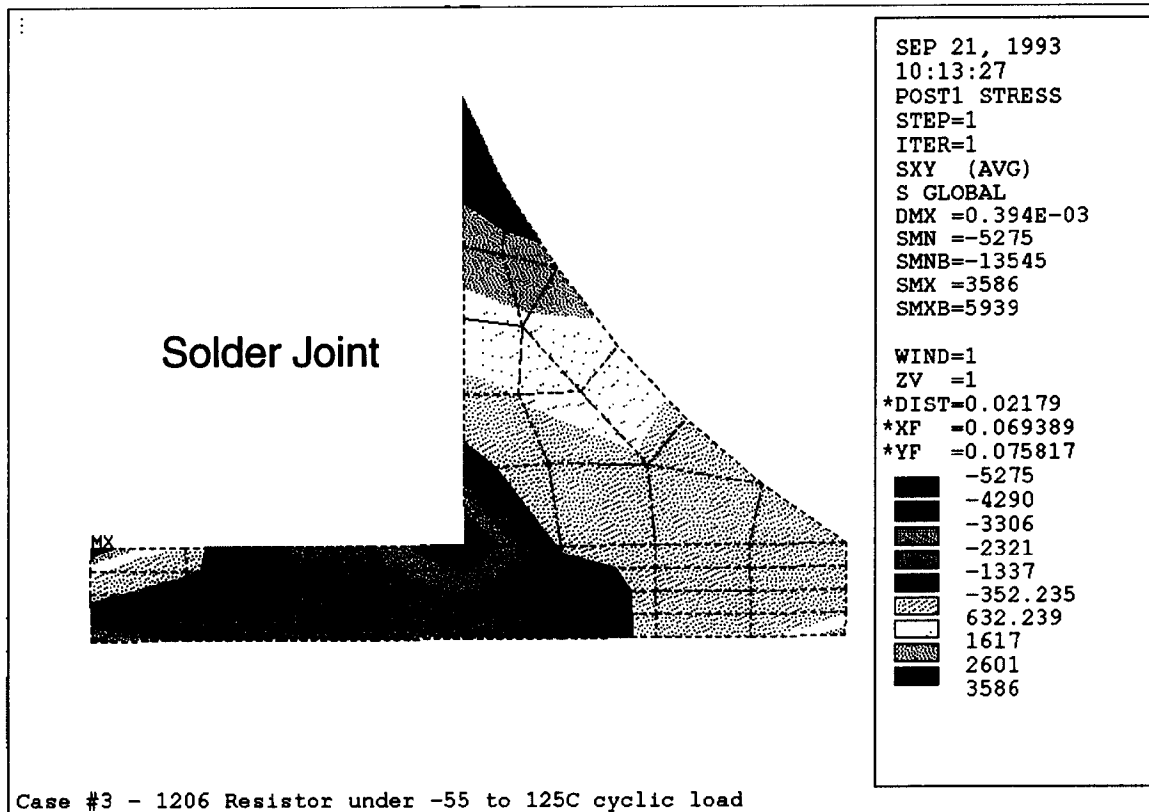


Figure 9.22 Shear Stress Distribution in Detailed Solder Joint (Case #3)

Example Parametric Study / Design Synthesis Scenario

As an example "what-if" design scenario, consider the case where the designer knows the target life the solder joint must meet and wants to see what other parameters (variables) can be changed to achieve that target life. In other words, "What if I want the fatigue life to be 40,000 cycles, and I let the solder joint height vary?" Thus, the target life becomes an input to the analysis (e.g., 20,000 or 40,000 cycles) and the parameter allowed to vary (e.g., solder joint height or PWB CTE) becomes the output.

Assuming one wants to vary parameters like the these examples just given, Figure 9.23 is an appropriate subsystem view of the SJTF Model for this scenario. Table 9.4 is the accompanying I/O Table that could be manually determined by tracing the extended

constraint graph in Figure 9.13. Note that this subsystem and I/O table are applicable equally to the case where either the Extensional Model is chosen as the Strain Model [Option 1.1], or the Plane Strain Model (assuming the corresponding outputs are supported by the FEA technique used) is chosen [Option 1.2]. This dual applicability is due to having the exact same between the SJTF Model and this subsystem.

Solder Joint Fatigue
Design Parameter Model

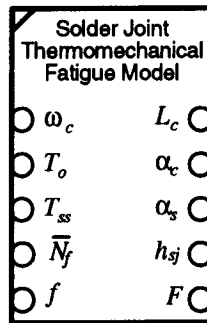


Figure 9.23 Example Subsystem View for SJTF Model

Table 9.4 Example I/O Table for SJTF Model

	T_o	T_{ss}	f	\bar{N}_f	ω_c					
						L_c	α_c	α_s	h_{sj}	F
1.	I	I	I	O	I	m	m	m	m	m
2.	I	I	I	I	I'	m	m	m	m	O
3.	I	I	I	I	I'	m	m	m	O	m
4.	I	I	I	I	I'	m	m	O	m	m
5.	I	I	I	I	I'	m	O	m	m	m
6.	I	I	I	I	I'	O	m	m	m	m

In the current prototype, only the Extensional Model case has been implemented using constraints. Therefore, it has been used to carry out this design scenario to achieve the results given in Table 9.5 using the indicated baseline values. I/O combination #3

allows solder joint height, h_{sj} , to vary, while #4 does the same for PWB CTE, α_s . Note that I/O combination #1 is the one used to obtain the baseline result of 25,062 cycles.

Table 9.5 Solder Joint Fatigue Parametric Study Using Implementation with Constraints

Baseline Values
PBAM: SJTF Model & Extensional Model (Level 1)
Conditions: Power Cycling, $T_o=20^\circ\text{C}$, $f=1$ cycle/day
PWA: PN 95415
Component Occurrence: R109, 1206 SMD resistor,
 PN 99120, $L_c = 0.125$, $\alpha_c = 6.7\text{E-}6$ (in/in)/ $^\circ\text{C}$, $T_c = 89^\circ\text{C}$
PWB: PN 99120, FR4, 0.062" thick, $\alpha_s = 15.0\text{E-}6$ (in/in)/ $^\circ\text{C}$, $T_s = 88^\circ\text{C}$
Solder Joint: 60Sn 40Pb solder, $h_{sj} = 0.010$ "

Case #11 Variation	\bar{N}_f (cycles)	h_{sj} (in)	α_s ((in/in)/ $^\circ\text{C}$)
#11.baseline	25062	0.0100	15.0e-6
#11.h.a	20000	0.0090	15.0e-6
#11.h.b	40000	0.0124	15.0e-6
#11. α_s .a	20000	0.0100	15.9e-6
#11. α_s .b	40000	0.0100	13.4e-6

Bold indicates output (result) for given Case #

As could be expected, results show that an increased fatigue life can be achieved by increasing the solder joint height, h_{sj} . For example, Case #11.h.b determined that the solder joint height should be 0.012" to achieve a desired life of 40,000 cycles. Alternately, an increased fatigue life also can be achieved by selecting PWB materials with lower CTEs, α_s . Note that fatigue life is quite sensitive to both these parameters.

If solder joint height is the desired output (I/O combination #3), the analysis result potentially can directly change the design due to the simple product-analysis transformation (PAT) involved in this case (where solder joint height is assumed to be proportional to the thickness of the stencil used during manufacture). However, even here

not just any stencil thickness can be chosen as they typically come in standard sizes. Also, too thick of a stencil can cause solder bridging during manufacture.

Thus, the focus in this thesis has been on getting the analysis result back to the point where it could be considered along with other variables in a design decision. In other words, the input to the PAT design synthesis operation can be obtained from the PBAM (assuming the constraint solver can obtain that variable as an output). Furthermore, once the design decision has been made, its impact can be rapidly accessed by re-running the same PBAM.

9.6 Discussion of Solder Joint Fatigue Case Studies

Strengths

1. Using the SJTF Model with the Plane Strain Model option [1.2] shows how a PBAM enables interaction of formula-based (Fatigue Model subsystem) and finite element-based analysis models. In the underlying constraint graph, the relations that the Plane Strain Bodies System contains are treated as any other relation. The fact that the relations require a finite element analysis solution is immaterial with respect to the structure of this PBAM. The interaction of this analysis model with other models in the SJTF Model constraint graph that have different solution methods also appears possible.
2. The parametric study example demonstrates how PBAMs can enable multidirectional analysis. Thus, analysis models can potentially interact in different directions, and "what if" design scenarios like that described can be supported.
3. Obtaining fatigue life from the SJTF Model using the Extensional Model option takes less than a few seconds, while using the Plane Strain Model option requires around four

minutes (depending on network and machine loads as well as selected options - some details are given in Chapter 10 under THESIS OBJECTIVE 4 (Speed)). Even with the Plane Strain Model option, only a few percent of the total time is spent creating the ANSYS PREP7 file and using the FEA results in the Coffin-Manson Model. Hence, in these cases PBAMs provide relatively rapid analysis results where the speed is limited by the solution procedure rather than by model creation.

4. The Plane Strain Model options (different stress-strain models and varying geometric detail) demonstrate how PBAMs allow flexibility in analysis model complexity.
5. The analysis results in Table 9.3 re-emphasize the need to check solder joint reliability since fatigue poses a potentially significant problem. For example, if your personal computer (PC) contains the PWA used in Case #61 (Figure 9.20) and you turn it on/off every day, the SJTF Fatigue Model with the Extensional Model predicts your PC will fail in less than 13 months of use. (Actually it would probably fail earlier since this number is the average predicted life of *only this component*. Since total system reliability is related to the product of the reliability of each component and each failure mode [Dieter, 1983], component reliability needs to be greater than the target system reliability). Since most electronic products do not fail in this predicted time frame, evidently PWAs such as this one have been designed (or redesigned) to have cooler component temperatures and/or to use components with smaller CTE mismatches. Thus, performing such analyses frequently and rapidly during design (which PBAMs enable) is seen to be helpful, if not essential, for product success.

Issues

1. For the same physical situation, the analysis results (Table 9.3) given by analysis models with different options vary quite a bit (e.g., \bar{N}_f in Cases #51 and #52 differs by

an order of magnitude). These discrepancies call into question how appropriate the analysis model options are. In particular the rectangular geometry option added in this research appears to have a stress singularity. Also, the linear solder model is not really valid for the cases where the stress exceeds the yield stress. However, these analysis models and added options still serve their purpose with respect to this research, as their variety of features demonstrates the flexibility of the PBAM representation. The fact that *a PBAM is only as good as the analysis model it represents* is, nevertheless, a very important point which is highlighted here.

2. Limitations on input/output combinations in general are discussed in Chapter 10. In brief, solution procedures must exist for each relation in the direction it will be run for a given I/O combination. Recall that some solution procedures have natural I/O combinations (THESIS OBJECTIVE 9 Directionality).

For example, in the Plane Strain Model FEA solution, variables such as component temperature are natural inputs while maximum solder joint stress is a natural output. Reversing the roles of these two variables would require a more expensive iterative solution procedure. Furthermore, some constraint solvers do not support I/O combinations requiring the solution of simultaneous equations - a situation that is required to obtain any one of the temperatures as an output in the SJTF Model.

3. The Plane Strain Model provides one example of how some information is missing in analysis model descriptions. The paper by Lau, et al. [1986] did not include the length of the PWB section, the initial load step, or convergence criteria (not that such detail should be included) Such cases make it difficult to reproduce an analysis model exactly.

9.7 PWA Warpage Case Study

9.7.1 Conceptual PBAM for PWA Warpage

The primary purpose of this case study was to test the ability of the PBAM representation to handle the interaction of global and local analysis models which are analysis models at regional levels (OTHER OBJECTIVE 16 Resolution). The warpage PBAM developed in this case study is shown in Figure 9.24, and is included in Appendix G. Note that this PBAM, **PWA Warpage Model**, has been developed at a conceptual level only and has not been implemented. The analysis model it represents would determine the deformations of the entire PWA using simplified analysis models of the components.

9.7.2 Use in PWA Global/Local Model

Conceptually, the interaction between global and local models (OTHER OBJECTIVE 16 Resolution) is the same as the interaction of two subsystems in a PBAM. Boundary conditions (as the name implies) are natural interface points between analysis models, from both physical and information exchange viewpoints. Some subset of results is extracted from the global model and fed into the local model as boundary condition inputs

Figure 9.25 demonstrates this concept for the **SJTF Model**, which is extended here to include warpage effects determined by the added subsystem that is of type **PWA Warpage Model**. Another option category has also been added to the **SJTF Model** (where Option [3.1] neglects warpage effects, and Option [3.2] includes them). Observations from this exercise are included in the next section.

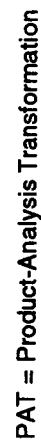


Figure 9.24 Conceptual PBAM for Analysis of PWA Warping

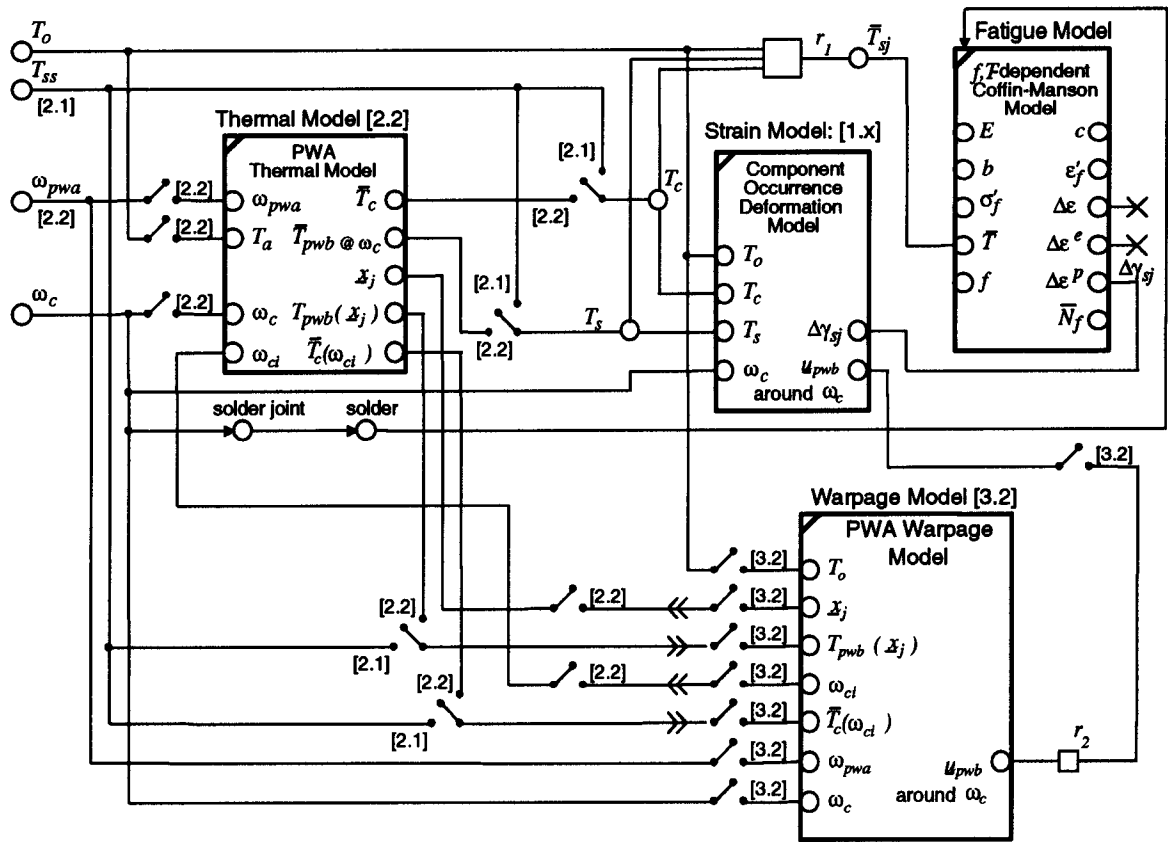


Figure 9.25 Study of Global / Local Analysis Model Interaction

9.7.3 Discussion of PWA Warpage Case Study

The primary observations from this conceptual case study are as follows:

1. All examples in the previous sections have **predefined compositional topology**, i.e., the number of bodies that compose the model is known *a priori*. For example, both the Level 1 and Level 3 models have four bodies (component, PWB, and two solder joints - but only one solder joint is modeled due to symmetry).

Often analysis models have **postdefined compositional topology**, where the number of bodies involved is not known until a specific product instance is selected.

This phrase means that the number of bodies modeled in the PBAM varies, not as an option value, but as determined by the assembly of the product itself. This situation occurs when the analysis model must include the effects of each member in a aggregate product variable. For example, the PWA warpage model in this section would typically have a different number of component occurrences for each different PWA analyzed, and different PWBs may have a different of layers. Currently the PBAM representation does not support postdefined compositional topology.

2. *Higher dimensional fields of state* often need to be exchanged between global/local models (e.g., 3D deformations along a closed path around the component of interest in the PWA Warpage Model would be input into the 3D Level 4 model). This issue raises other issues:
 - a. The global subsystems must have operators that support the output of such fields (and input for inverse problems).
 - b. If discretization solution techniques are used (e.g., FEA), the local model needs to tell the global model for which discrete points it needs values. This two-way interaction may need to occur even though the overall result is a one-way analysis, as can be seen in the work by Niu and Shephard [1990].
3. The PBAM must support *mappings between different coordinate systems and different compositional topology* in the global/local models. This ability effects Issue 2 and has not been addressed explicitly. Coordinate systems probably need to be an attribute in each ABB, in which case coordinate transformations between ABBs could potentially be represented as relations.
4. PBAMs are extendible with apparent relative ease due to their modularity (demonstrated here at least on a conceptual PBAM development level). Figure 9.25

illustrated the relatively small impact adding warpage effects had on the structure of the original solder joint fatigue PBAM.

Preliminary work is promising and indicates that PBAM partitions (the constraint subgraphs defined in Chapter 6 that enable representation of analysis model options) may be part of the solution to the postdefined compositional topology problem (Item 1). Item 2 can be handled by the current PBAM structure if one views such fields as a variable that happens to be a relation.

The basic idea of Item 3 also appears doable if operators exist and the inter-system mapping is not too complicated. It is worth noting that Item 3 emphasizes the need for something like a PBAM to help subsystem models interact. *The PBAM has access to the analysis context (product entities, analysis entities, and options) which is the common source for all the information each subsystem model needs.* The PBAM also knows what the purpose of the interaction is, and thus shows promise for enabling more general global/local interaction (OTHER OBJECTIVE 16 Resolution).

In summary, the conceptual PWA warpage case study brought out both strengths and weaknesses in the present PBAM representation. Generalized global/local interaction and postdefined compositional topology are not currently supported in the PBAM representation.

9.8 Summary

In summary, it is felt that developing and implementing PBAMs for these analysis models has helped validate the PBAM representation, has illustrated how to develop and implement PBAMs, and has demonstrated the usefulness of PBAMs through representative design scenarios.

CHAPTER 10

EVALUATION

The most important figures are unknown and unknowable.

W. Edwards Deming

[Walton, 1986, p. 36]

10.1 Evaluation Approach

As noted by Sargent [1985], validation and verification of a model is one of the more critical issues in the modeling process. He gives the following definitions which distinguish between the two terms:

validation substantiation that a computerized model within its domain of applicability possesses a satisfactory range of accuracy consistent with the intended application of the model.

verification ensuring that the computer program of the computerized model (i.e., the simulator) and its implementation is correct.

His figure depicting the various forms of a model and the various types of validity and verification is reproduced here for reference along with his explanation:

The *problem entity* is the system (real or proposed), idea, situation, policy, or phenomena to be modeled; the *conceptual model* is the mathematical/logical/verbal representation (mimic) of the problem entity developed for a particular study; and the *computerized model* is the conceptual model implemented on a computer. The conceptual model is developed through an *analysis and modeling phase*, the computerized model is developed through a *computer programming and implementation phase*, and inferences about the problem entity are obtained by conducting computer experiments on the computerized model in the *experimentation phase*.

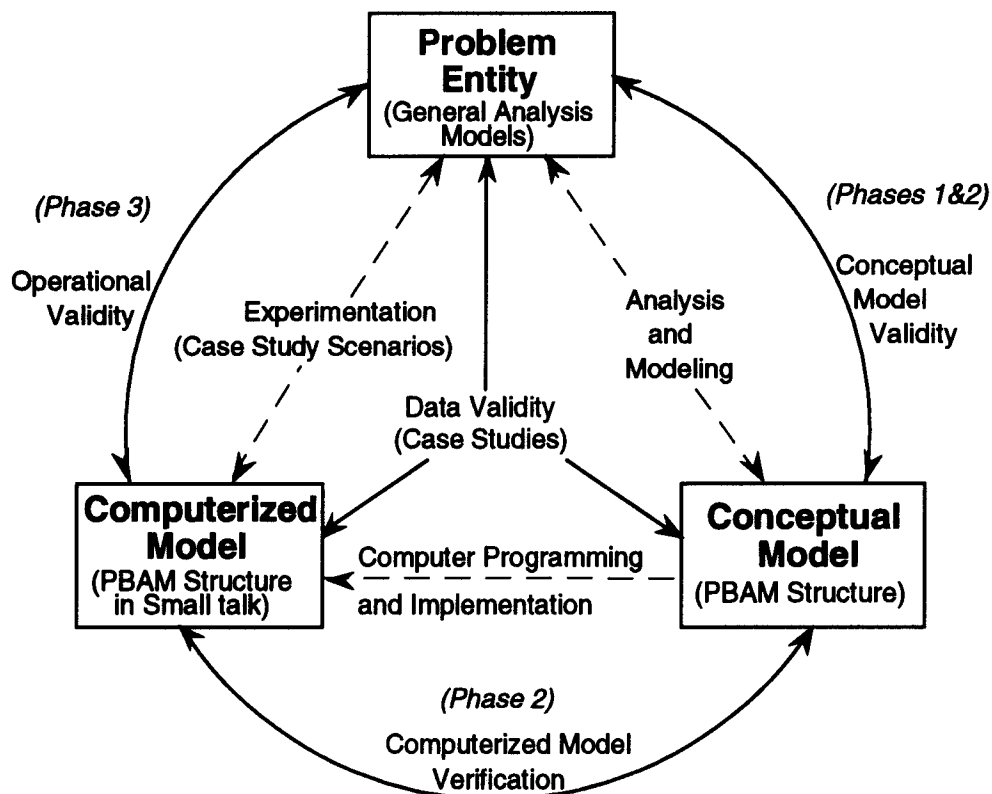


Figure 10.1 Simplified Version of Modeling Process [Sargent, 1985]

We relate validation and verification to this simplified version of the modeling process as shown in Figure [10.1]. *Conceptual model validity* is defined as determining that the theories and assumptions underlying the conceptual model are correct and that the model representation of the problem entity is "reasonable" for the intended use of the model. *Computerized model verification* is defined as ensuring that the computer programming and *implementation* of the conceptual model is correct. *Operational validity* is defined as determining that the model's output behavior has sufficient accuracy for its intended application. Data validity is defined as ensuring that the data necessary for model building, model evaluation and testing, and conducting the model experiments to solve the problem are adequate and correct.

While Sargent emphasizes the validation and verification of industrial engineering-type simulation models, Ignizio [1991] defines the following phases of validation and verification with a focus is on general rule-based expert systems:

Phase 1 The justification for the employment of an expert system.

Phase 2 The validation of the *consistency* and *completeness* of the expert system's rule base.

Phase 3 The verification of the overall *performance* of the expert system.

Labels have been added to Figure 10.1 that show the rough correspondence between Ignizio's Phase 2 with Sargent's Conceptual Model Validity and Computerized Model Verification and between Phase 3 and Operational Validity. Ignizio's Phase 1 could be considered part of Theoretical Validity.

Before the research of this thesis can be validated, one must recall exactly what the "model" is that is being validated. As stated in Chapter 2, a PBAM is a representation (an information model) of analysis models that includes linkages to a product model - that is, the PBAM representation is a model of models. Hence, *the general PBAM representation is the "model" being evaluated in this chapter*. Therefore, the primary thing that needs to be determined is whether or not the general PBAM representation is a valid model of analysis models. This case is analogous to determining if the rule-based knowledge representation is valid rather than determining if a given populated rule base is valid for a particular application.

One part of demonstrating PBAM validity is to check if valid models of analysis models can be developed and implemented. Checking the theoretical validity of the analysis models themselves is beyond the scope of this thesis; however, checking that PBAMs of the analysis models give the same analysis results as other implementations is part of validating this research. Though Sargent and Ignizio have focused more on the

validation and verification of models rather than on models of models, it is felt that the process is very similar for both cases. The validation approach taken in this research is as follows:

1. Determine the objectives.
2. Determine the degree that PBAMs meet those objectives.
3. Compare the degree that PBAMs meet those objectives versus other approaches.

Implicit in the first step is that the objectives themselves are "good" objectives. From a research point of view, in this thesis "good" means that meeting the objectives will a) create new knowledge, and b) meet some engineering need. For knowledge to be considered "new," the concepts must provide either some capability not currently available or some significantly improved capability. Ignizio's Phase 1 is related to these criteria and has been executed by characterizing the problem as one which can benefit from AI techniques and by searching the literature for existing solutions. Furthermore, Chapter 4 identified what the objectives of this research are and Chapters 2, 3, and 4 showed that the objectives meet criteria a) and b). Therefore, Step 1 has been completed.

Step 2, determining how well the objectives have been met, has been discussed some in various parts of the thesis, but will now be consolidated and discussed in one section.

10.2 PBAMs Versus Objectives

Tables 10.1 and 10.2 summarize the degree that each THESIS OBJECTIVE and OTHER OBJECTIVE was met as demonstrated in the case studies. The reader is referred back to Chapter 4 (Objectives for Analysis Model Representations) for descriptions of each objective itself. The degree the PBAM representation meets each of the THESIS

OBJECTIVES (1-9) is addressed individually. Several key OTHER OBJECTIVES are then discussed. The degree to which each claim made in this chapter has been demonstrated will be denoted as follows:

Theoretical Development (TD) The concept was defined or demonstrated using logical, mathematical, algorithmic, and/or information modeling techniques.

Case Study Development (CSD) The concept is backed by TD and also was demonstrated in the case study development using the PBAM structure. In other words, one or more PBAM views were created of a representative analysis model which demonstrates the concept.

Case Study Implementation (CSI) The concept is backed by CSD and also was demonstrated in a computer implementation. Representative data sets were used, and example operations were performed that demonstrate the concept.

Table 10.1 Demonstration of THESIS OBJECTIVES in Case Studies

Objectives	Case Study PBAMs			
	Solder Joint Fatigue Model with Extensional Model	Solder Joint Fatigue Model with Plane Strain Model	Solder Joint Fatigue Model with Plane Strain Model and PWA Warpage Model (CSD)	
1. Representation	Yes	Yes	Yes	Yes
2. Routine Analysis Automation	Yes	Yes	Yes	Yes
3. Product-Analysis Associativity	Yes	Yes	Yes	Yes
4. Complexity Level	Low - Simple SJ Geometry - Linear Solder - Extension/Pure Shear	Medium, Moderately High - Simple/Detailed SJ Geometry - Linear/Nonlinear Solder - Plane Strain	Moderately High - Simple/Detailed SJ Geometry - Linear/Nonlinear Solder - Plane Strain	
5. Options	Yes	Yes	Yes	Yes
6. Modularity and Seamlessness	Yes	Yes	Yes	Yes
7. Speed	Fast (~1 sec)	Slower (~4 min.)	Slower	Slower
8. Flexibility	Yes	Yes	Yes	Yes
9. Directionality	Yes	(Yes)	(Yes)	(Yes)

Table 10.2 Demonstration of OTHER OBJECTIVES in Case Studies

Objectives	Case Study PBAMs			
	Solder Joint Fatigue Model with Extensional Model	Solder Joint Fatigue Model with Plane Strain Model	Solder Joint Fatigue Model with Plane Strain Model and PWA Warpage Model (CSD)	
10. Final Results	Yes	Yes	Yes	
11. Intermediate Results	Yes	Yes	Yes	
12. Interaction	Yes	Yes	Yes	
13. Multivalued Inputs	No	No	No	
14. Multivalued Outputs	No	No	No	
15. Solution Procedure Control	Not Applicable	Yes	Yes	
16. Resolution	Not Applicable	Not Applicable	(Global/Local Models)	
17. Model Exchange	Yes	Yes	Yes	
18. Existing CAE/CAD Tools	(Mentor Graphics)	(Mentor Graphics), ANSYS	(Mentor Graphics), ANSYS	
19. Encapsulation	No	No	No	

Table 10.2 Demonstration of OTHER OBJECTIVES in Case Studies (continued)

Objectives	Case Study PBAMs			
	Solder Joint Fatigue Model with Extensional Model	Solder Joint Fatigue Model with Plane Strain Model	Solder Joint Fatigue Model with Plane Strain Model and PWA Warpage Model (CSD)	
20. Purposes	Design Verification, Parametric Studies	Design Verification, Analysis Support	Design Verification, Analysis Support	
21. Product Domains	PWA	Same	Same	
22. Product Type	Leadless Component; Resistor/IC	Same	Same	
23. Applications	Comp. Occ. Deformation Solder Joint Reliability	Same	Same plus: PWB Warpage	
24. Disciplines	Thermomechanical	Same	Same	
25. Behavior Regimes	Linear, Steady State, Power/Thermal Cycling	Same plus Nonlinear	Same plus Nonlinear	
26. Solution Methods	Formula-based	Same plus Finite Element-based	Same plus Finite Element-based	
27. Variables	Entities, Reals, Integers	Same	Same plus collections	
28. Relations	Discrete, Continuous; Equality; Linear, Nonlinear	Same	Same	
29. Systems of Relations	Nonlinear, Fixed Topology No Simultaneous Solutions	Same	Same plus Postdefined Compositional Topology	

10.2.1 PBAMs Versus THESIS OBJECTIVES

OBJECTIVE 1 Representation Provide a representation of analysis models which meets all other THESIS OBJECTIVES and many of the OTHER OBJECTIVES.

First, the PBAM representation will be evaluated against the previously given criteria (Chapter 3) that a good representation should fulfill.

- a. *Have a well-defined structure composed of a pre-defined vocabulary of symbols.*

In Chapters 5 and 6, the PBAM structure is defined, as well as the symbols that composed it (e.g., analytical and product variables, linkages, subsystems, and options).

- b. *Have a defined method for being developed from the specific real world entities being represented.* Preliminary guidelines were given in Chapter 7 on how to develop a PBAM of a specific analysis model given a description of that model as input. How good these guidelines are remains to be seen as other engineers attempt to use them.

- c. *Be capable of expressing any relevant aspects of the content of the real world input.*

This claim is rather difficult to substantiate unless one has tested the representation against "all" variations of real world input that exists. Therefore, the claim is made that the PBAM representation can represent a relevant class of engineering analysis models, but not all analysis models. Hence, the PBAM representation is complete and general within that class of analysis models, but it is almost certain that new constructs will need to be added as models beyond that class are considered.

- d. *Be easily modified.*

This capability is addressed in OBJECTIVE 6 (Modularity / Seamlessness).

- e. *Be understood by the people who are familiar with the knowledge being represented.*

No attempt has been made to formally demonstrate fulfillment of this criteria (e.g.,

through field trial with students or practicing engineers). Nonetheless, in discussions with several students, faculty, and industry engineers, there at least appears to be a high-level understanding of what a PBAM is and how to interpret a PBAM of a particular analysis model.

- f. *Be explicit and transparent.* This criteria is met because many important features of an analysis model (e.g., variables, relations, and PATs) are readily identified in the PBAM representation. In contrast, a neural network representation of analysis models would be considered an implicit representation where such features would be distributed among all the nets and, thus, would be difficult to access and identify.
- g. *Hide detail unless expressly requested (i.e., support encapsulation).* One way that PBAMs satisfy this criteria is by being composed of other structures that can themselves be composed of other structures. Also, as evidenced by subsystem views of a PBAM and accompanying I/O table views, all one must do to use a PBAM is plug in the appropriate inputs and select desired options. One need not know all the internal details.
- h. *Be concise.* The PBAM structure meets this condition with the exception of some redundancy as noted (Chapter 5) which is there for convenience (as are all the other views defined).
- i. *Be computable.* Preliminary guidelines for implementing PBAMs have been given in Chapter 8 and PBAMs of actual analysis models have been prototyped. General ABB operations and associated algorithms still need to be defined; preliminary results towards this goal are given in Chapter 5. As noted in the aforementioned chapter, more experience with other constraint solvers and more analysis models is certain to improve these guidelines.

As mentioned in Chapter 9, the prototype does not completely match the final PBAM and ABB representations of the case study models. Similarly, PBAM options, subsystem substitution, and nested PBAMs have not been implemented using constraints. However, two GPABBs (Coffin-Manson Model and Interconnected Rods System) have been implemented successfully as subsystems in a PBAM (the PWA Two Rod Model - the precursor of the Extensional Model) using constraints.

Generally, these concepts have been implemented to some degree at least in non-constraint-based form. Also, sufficient other constraint-based implementation was done to give reasonable certainty that the constraint-based implementation of these concepts would be reasonably close to that expected.

- j. *Be efficient.* As will be seen in the discussion for OBJECTIVE 7 (Speed) below, the case study PBAMs executed reasonably quickly. Speed performance with large numbers of PBAMs attached to many product models has not been tested..

How much memory PBAMs use has not been determined. Especially in the case where relations requiring numerical solutions are involved, memory usage could become a factor just for storing all the PBAM instances that could be used on a single design (e.g., some PWAs have over 700 components). Therefore, unless necessary for audit purposes, a strategy for keeping only essential results or a policy of completely discarding used PBAM instances and re-executing them as necessary could be instituted.

Having met most of the above criteria to a fair degree, the PBAM representation at least qualifies to be called a decent knowledge representation. Presently, it is the only known representation of analysis models that supports all these capabilities. Also, as will be discussed in the remainder of this section, and as summarized in Tables 10.1 and 10.2, PBAMs have many, but not all, of the capabilities defined in both the THESIS OBJECTIVES

and OTHER OBJECTIVES. Therefore, the claim is made that the PBAM representation fulfills this objective to a reasonable degree.

OBJECTIVE 2 Automation Fully automate routine analysis.

PBAMs do not *fully* automate routine analysis in that the analysis context must perform fully the routine analysis process steps marked with an asterisk (*) in Figure 10.2 (repeated here for convenience from Chapter 5) as demonstrated in the case studies in Chapter 9. The analysis context must perform fully those steps marked with a dagger (†) and, in most cases, must assist partially or fully the step marked with a double dagger (‡).

1. Design product.
2. Perform routine analysis.
 - 2.1 Identify application (the design problem).
 - 2.2 Choose ABB for application.
 - † 2.3 Setup ABB.
 - † 2.3.1 Instantiate ABB.
 - † 2.3.2 Specify options.
 - † 2.3.3 Link ABB with input/output entities.
 - † 2.3.4 Specify I/O combination.
 - † 2.4 Reconcile ABB.
 - * • Link analysis model(s) with product & analysis inputs.
 - * • Execute (solve) analysis models(s).
 - * • Manage interaction of analysis models.
 - * • Extract results.
 - † 2.5 Read results.
 - 2.6 Check results.
 - 2.7 Interpret results to determine design changes.
- ‡ 3. Make design changes.

Figure 10.2 Routine Analysis Steps Automated Using PBAMs

Chapter 5 defined the operations a PBAM does automatically, as well as those the analysis context must perform. In Step 3 above, the analysis context need only connect the

product and analysis entities at a relatively high-level. For example, as demonstrated in Chapter 9, the analysis context connects a PWA *object* to the PBAM rather than manually inputting all the dimensions and material properties into the PBAM - the PBAM extracts such information automatically from the PWA object.

OBJECTIVE 3 Associativity Link analysis models with product models.

The PBAM representation supports information flows in both directions: from product model to analysis model and from analysis model to product model. The modeling idealizations and design tasks that occur in these directions, respectively, have been characterized (Chapter 3). This research has focused more on modeling idealizations than on design synthesis operations (in order to focus on supporting the design verification task). This thesis does not claim to have developed all such transformations, but does claim that the structure provided supports the following classes of transformations:

- a. Modeling idealizations that can be formulated as relations with product variable inputs and analytical variable outputs (CSI, Chapter 9).
- b. Design synthesis tasks that can be formulated as relations that require only analytical variable inputs (e.g., geometric sizing) (CSD). As demonstrated in the design synthesis scenario in Chapter 9 (CSI), such tasks can be performed to the extent that at least the analytical variables can be obtained which are inputs to the design operation (assuming the relations allow such inversion). Design operations requiring product variables not used in the PBAM have not been addressed in this thesis, but may be possible if they are accessible through the product variables connected to the PBAM.

Unfortunately, PATs in general are not as well defined or characterized as other types of analysis relations. This situation was encountered in the case studies where some design-

analysis linkages had to be inferred from the papers by Engelmaier and by Lau, et al. Also, from an analysis model representation viewpoint, it is hoped that generalized PATs can be developed (e.g., transformations to determine the material in a multi-material fabrication which dominates a certain behavior - like the dominant CTE PAT in the Extensional and Plane Strain Models in Chapter 9).

The intent of this thesis has not been to develop generalized PATs; *the intent of this thesis in this respect is to show how some linkages between product and analysis models can be represented if they are formulated as relations*. Then, as more generalized PATs become available, they can be used in the current PBAM structure. Observations were given in Chapter 3 to characterize what types of product-analysis transformations PBAMs must support in order to link a product model with an analysis model.

One final observation is that *linkages to the product model can enable modular analysis model complexity variation* (OBJECTIVE 4) because each analysis model has access to the product model to extract the information it needs. This situation is seen clearly in the SJTF Model (Chapter 9) which uses the exact same interface to the Strain Model subsystem both when that subsystem is the Extensional Model and when it is the Plane Strain Model. It is only inside these subsystems that it is apparent that the latter PBAM uses more product information (and is more complicated) than the former.

OBJECTIVE 4 Complexity Level Represent analysis models of varying complexity (e.g., topology, geometry, material model) for the same regional resolution.

This new capability was demonstrated in the case studies (Chapter 9) by representing models with varying degrees of complexity in the following area:

Body Complexity

- Deformation Behavior (PWB/Component): extensional (1D) (CSI), plane strain (2D) (CSI)
- Deformation Behavior (Solder Joint): pure shear (1D) (CSI), plane strain (2D)(CSI)
- Material Model (Solder): linear elastic (CSI), bilinear elastoplastic (partial CSI)
- Geometry (Solder Joint): 1D height (CSI), 2D rectangle (CSI), and 2D parametric shape (CSI).

Compositional Complexity

Topology (PWB): homogeneous body (CSI), multilayer laminate (partial CSD)

(Postdefined compositional topology is not supported, per the discussion in Ch. 9).

Therefore it is claimed that PBAMs at least can represent multiple models that vary in complexity along these dimensions.

OBJECTIVE 5 Options Allow choices in analysis model operation.

Representing alternative model operation within one model representation is an integral part of the PBAM structure as defined in Chapter 5 for the ABB representation. The following types of options have been demonstrated (Chapter 9):

- **Model Complexity:** Behavior (extensional and plane strain), Material model (linear and nonlinear solder model), Geometry (rectangular and detailed solder joint) (CSI)
- **Load Type:** Thermal vs. power cycling (CSI)
- **Effects:** Warpage effects in solder joint fatigue (partial CSD)

Options and degrees of complexity (OBJECTIVE 4) are related in that options allow degrees of complexity to be packaged in one PBAM - the user simply selects the options desired.

To date options must be chosen before the PBAM is initialized so that if an option change is desired, the PBAM must be re-initialized (CSI). Changing options in a constraint-based implementation after initialization looks promising (as it involves deleting and adding constraints which are standard constraint graph operations), but presently no further claim is made with regard to changeability.

OBJECTIVE 6 Modularity/Seamlessness Provide modularity to represent new analysis models with minimal impact on analysis models already represented.

The following types of modularity have been demonstrated:

PBAM Addition New PBAMs were added in Chapter 9 with minimal impact on existing ones. Adding the Plane Strain Model could have been achieved with no changes at all to the Extensional Model; however, due to the commonality between the two PBAMs, parts of the Extensional Model can be shifted into a new PBAM superclass, Component Occurrence Deformation Model, which decreases the effort required to add the Plane Strain Model. Therefore, such changes are viewed as good and have little negative impact on the pre-existing PBAM (CSD/I).

PBAM Change Both PBAMs and analytical systems were changed with little structural impact to accommodate inclusion of warpage effects in the calculation of solder joint fatigue (CSD, Chapter 9). Also one can see how changing the SJTF Model to include power cycling if that were not considered from the beginning (as in the case of the Plane Strain Model) could be readily accommodated (CSD)

Constructive Modularity PBAMs can be constructed modularly from analytical building blocks (CSI) including other PBAMs (CSD). This approach of building larger entities from smaller ones is what localizes and minimizes the effects of addition and change just discussed.

Subtraction of PBAMs and of features from PBAMs has not been addressed.

OBJECTIVE 7 Speed Give rapid results.

This objective is performance related and thus falls under Ignizio's Phase 3 and Sargent's operational validity categories. Table 10.3 compares total wall clock run times of various case numbers (see Table 9.3) under the following computing conditions:

1. VisualWorks 1.0 was running under Sun OS 4.1.2 on Sun SPARCstation2.
2. ANSYS 4.4a was running under VMS on a super scalar VAX 6610 with 250 Mb of RAM and a 245 Gb hard disk.
3. File transfers between the VAX and SUN were done automatically via the Georgia Tech Network which is a TCP/IP network. Its transfer rate varies depending on network load.

Table 10.3 Case Study Total Execution Times

Case # (SJTF Model plus indicated Strain Model and Options)	Implementation with Constraints	Implementation without Constraints	
		Initial Setup	Post-Setup
#1 Extensional Model	< 1 sec	~ 3 sec	< 1 sec
#2 Plane Strain, Linear Solder, Rectangular SJ	257 sec (4.3 min.)	NA	NA
#3 Plane Strain, Linear Solder, Detailed SJ	~ 5 min.	NA	NA

For Case #2 (see Table 9.3), which required file transfers and solution via ANSYS on the remote VAX, the total time can be broken down as follows:

Table 10.4 Breakdown of Execution Time

	Time (sec)	% of Total
Setup and PREP7 input creation	3	1%
PREP7 file creation and transfer to VAX	39	15%
ANSYS execution (PREP7, solver, POST1)	198	77%
Solution file transfer to SUN	8	3%
Solution file parsing	8	3%
Final Calculations	1	<1%
Total	257	100%

As can be seen from this breakdown, the first step (FEA input file creation) and last step (results transfer and fatigue model creation/calculation) take only a small percentage (~2%) of the total time. Similar results were observed for the other cases involving finite element-based models. Thus, a reasonable rule of thumb (at least for constraint graph solutions that do not require simultaneous equations) is that the total solution time would be comparable to (but slightly greater than) the total time that would be required by a non-constraint implementation. This approximation becomes better with increasing numbers of individually time-consuming relations.

It is unclear how applicable this observation would be to other PBAMs with general constraint graphs, but it appears that by far one "expensive" constraint (like one that requires a finite element based solution) is more costly time-wise than propagation across numerous "cheaper" constraints. This story would probably change for PBAMs requiring simultaneous equation solution in a constraint graph with non-linear constraints; however, even then if the expensive constraints were in the simultaneous equation loop, it seems they would tend to dominate the total calculation time.

These case study results point to the following tentative observation: *PBAMs are subject to the expected trade-off between efficiency and flexibility.* Constraints are a rather natural representation of analysis relations that, when implemented using constraint solvers (e.g., DeltaBlue), can ease implementation and offer multi-directional capabilities; however, implementation of these constraints using non-declarative forms (e.g., procedural methods in straight Smalltalk or C++) can potentially achieve greater performance by capitalizing on specialized forms (that then are typically harder to extend and maintain).

OBJECTIVE 8 Flexibility Accommodate a wide variety of analysis models.

Here the degree of flexibility is assumed to be related to the degree that OBJECTIVES 20-29 are met as a whole (Chapter 3). One can see from Table 10.2 that each of these objectives except 21 (Product Domains), 22 (Product Type), 24 (Disciplines), and 29 (Systems of Relations) were demonstrated by at least two different values in the case studies. These demonstrations, coupled with the below discussions of the indicated exceptional objectives are the basis upon which the qualitative claim of reasonable flexibility is made.

OBJECTIVE 9 Directionality Allow different combinations of inputs/outputs.

Directionality capabilities are summarized here and are largely limited by existing constraint solving algorithms (if the PBAMs are so implemented) and by existing relation solution procedures, rather than the PBAM representation itself. In the rest of this item, the "constraint graph" refers to the PBAM constraint graph that results after all options have been selected.

The following conditions must be satisfied first by all relations in the constraint graph when the relation is looked at *individually* (i.e., when it is disconnected from the graph).

CONDITION 1 Each relation must be individually implementable in a computer to the following degree: Each variable that will be requested as an output of the relation must have a corresponding solution method

In other words, it is assumed the constraint graph solver will not ask a relation for an output it inherently cannot give because no solution procedure exists. This objective in Chapter 3 discussed how some solution procedures (e.g., FEA) have natural inputs and outputs, and that requesting a natural input as an output may require increased computational cost if the solution is possible at all. Therefore, in this research only variables that can be determined by existing solution procedures are available for output in an individual relation. The responsibility of how the inverse is performed for a particular solution procedure is left to the developers of such procedures. The discussion of this objective in Chapter 4 gave one example of how ANSYS provides such capabilities in the case of finite element analysis.

In addition, the following condition is assumed to be satisfied as the impact of cases beyond this condition has not been resolved in this research.

CONDITION 2 Given corresponding single-valued inputs, the available outputs for each relation is single-valued.

Once these two Conditions are met, whether or not the constraint graph can be solved by existing algorithms depends on several factors including the topology of the graph, the types of relations and variables, and the desired I/O combination. Algorithms are known

to exist for at least the following cases, again keeping in mind that the above conditions are met as follows:

CASE 1 If the constraint graph contains no cycles, the greatest flexibility exists on allowable types of relations, variables, and I/O combinations, including the following non-exhaustive examples:

Relations: Discrete/continuous, linear/nonlinear, logical, equality

Variables: Real, integer, logical, entity

I/O Combinations: All fully constrained combinations

DeltaBlue [Freeman-Benson, et al., 1990; Maloney, 1991] can handle this case, as was demonstrated in Chapter 9 for the design synthesis scenario.

CASE 2 If the constraint graph contains cycles, and the desired I/O combination does not require the solution of simultaneous equations, then the types of relations and variables listed in Case 1 are also allowable.

No statement is made for constraint graphs that fall outside these two cases, though some algorithms exist for more general cases [Freeman-Benson, et al., 1990].

10.2.2 PBAMs Versus OTHER OBJECTIVES

As the above objectives were the major focus of this research, only some of the remaining OBJECTIVES (10-28) will be discussed briefly. Table 10.2 shows how many of the remaining objectives are met to at least some degree.

OBJECTIVE 14 Multivalued Outputs Allow multivalued output variables.

This situation would occur not only when multivalued inputs are given (just described) but also when a relation involved in the analysis has multiple solutions (e.g., $x = \pm y$).

When a relation has non-unique inverses (i.e., multivalued outputs), the question arises as to how they would propagate through the constraint graph. One simple example of such a relation is the absolute value operation in the Solder Joint Fatigue PBAM in Chapter 9 (Eqn. 9.8). An "operand list" looks promising here. The relation would create an operand list containing each solution as an item in the (ordered) list. When the operand list is passed to the next relation, operators in that relation would be performed on each individual member in the operand list. However, this idea has not been tested. The current approach to such relations is to implement them such that only one value is returned (in the case study, only the positive value is available for the inverse form of Eqn 9.8).

OBJECTIVE 19 Encapsulation Encapsulate existing tools that have specialized design-analysis linkages.

This objective has not been demonstrated in this research. It is believed that it can be met if the existing tools can be modeled as relations, and if the use of these relations fall under CASES 1 or 2 defined in OBJECTIVE 9 (Directionality) above. The variables in such relations would be both product variables and analytical variables. The degree to which these relations can be integrated for automated PBAM solution depends on at least two factors: whether or not the tools 1) support programmatic entry of these variables, and 2) support programmatic control of themselves.

OBJECTIVE 21 Product Domain Be usable in multiple engineering product domains.

The major assumption about the product model structure with respect to its use in the PBAM structure is that it is compatible with a object-like structure (entity, attribute (part-of relation, where complex entities are allowable attribute values), and is-a relation) like that defined in the EXPRESS modeling language [ISO 10303-11]. Use of such a product model was demonstrated in the case studies (Chapter 9).

Rather than being restricted to the EXPRESS notation of derived attributes, it is assumed that general relations (constraints) can exist among attributes. These relations also are subject to CONDITIONS 1 and 2 in OBJECTIVE 9 if they are meant to be active in the PBAM constraint graph. In other words, if a product can be modeled using EXPRESS and such relations, it is compatible with the PBAM structure. Therefore, because the ISO STEP project is using EXPRESS to define product models for a wide variety of product types (including ships, PWAs, sheet metal parts, general assemblies, etc.), it is claimed that the PBAM structure is compatible with a wide variety of such products and OBJECTIVE 21 (Product Domain) has been met.

OBJECTIVE 24 Discipline Represent analysis models from different engineering disciplines. The mathematical similarities in the governing equations of various engineering disciplines have led to well-known analogous counterparts between disciplines and, more recently, to generalized terms for these counterparts in the bond graph representation [Rosenberg and Karnopp, 1983; Ingram and Masada, 1989 a & b]. The case studies and examples done in this research (Chapter 9) fall under the classification of thermomechanical models which are basically boundary value problems (BVPs) having the following general categories of variables and relations: bounded media, kinematic relations, constitutive relations, boundary conditions, and state variables.

As these categories were treated in the PBAM representation with a strong degree of generality, it is argued that PBAMs can represent analysis models that are BVPs involving continuous bounded media (e.g., conductive heat transfer). The fact that PBAMs with relations requiring finite element-based solutions were demonstrated helps confirm the claim, as a wide variety of such problems can be solved using FEA. Whether or not the present PBAM representation can handle problems with unbounded continuous media (such as fluids) has not been addressed.

As lumped parameter problems are typically structurally simpler than BVPs, it is felt that the PBAM representation can support analysis models involving lumped parameter systems since it can handle boundary value problems with bounded media; therefore, OBJECTIVE 24 (Discipline) has been met.

10.3 General Discussion

This section discusses points about the PBAM representation that are more global in nature than the preceding objectives. The two sections after this one discuss other such points in detail.

1. The case studies described in Chapter 9 involve geometry that is relatively simple and can be parameterized. This limitation has been necessitated primarily by limited prototype capabilities which transform a general purpose analytical system into a FEA input file (capabilities which are not the focus of this thesis). Similarly, the information exchanges between subsystems have been single discrete values (versus a time- or space-varying field of discrete or continuous values).

It is felt that the main impact increased geometric complexity will have is the need for more complex product-analysis and analysis-analysis transformations. The

current PBAM *structure* can already support such new transformations since it would represent them in the same way as any other relation. However, it is acknowledged that other unforeseen factors may impact the PBAM representation in this respect. unknown.

3. Analysis model descriptions may be difficult to piece together and typically may not contain all the information required to develop PBAMs. For example, the paper by Lau, et al. [1986] did not include the length of the PWB section, the initial load step, or convergence criteria (not that such detail should be included). When it does exist, analysis model documentation typically suffers from a lack of product-analysis transformation intent. These situations may make it difficult to reproduce an analysis model exactly, but an experienced analyst can probably fill in the gaps to a sufficient degree.
4. "Routine analysis models" may not be easy to identify for newer disciplines such as PWA thermomechanical analysis. In fact, many analysis models that can be found in the literature may not have been originally intended to support product design directly. Simply defining the term "routine analysis model" spurs the notion that such models *should* exist (and they often do in mature product domains). Therefore, the existence of an analysis model representation like PBAMs could provide a tangible target for engineering analysts to aim at when developing new analysis models to support product design.
5. Similarly, the need for routine analysis models can direct further research and development to fill in gaps that may exist. As seen in Section 9.2 with respect to determining solder joint strain, existing analysis models may not consider some analysis variations or product variations of interest (e.g., components with epoxy dots or conformal coating). Thus, the case studies illustrate how the search for "routine"

analysis models can help identify areas requiring further *analysis model* development (which potentially could be added to the PBAM representing that analysis model when the analysis model extensions are mature). Once an existing routine analysis model has been identified or a new one has been developed, PBAMs can help maximize the use of that model during product design.

10.4 Potential Sources of Discrepancies

This section identifies possible reasons 1) why different PBAMs might be developed inadvertently to represent the same analysis model, and 2) why results from non-PBAM implementations of an analysis model might vary from those of a PBAM implementation. Reasons 1 and 2 cause discrepancies in the PBAM structure that represents an analysis model. Reasons 3 and 4 cause implementation and test differences.

1. *Incomplete analysis model description.* All of the relations may not be defined in the analysis model description from which the PBAM is developed. For example, the **Plane Strain Model** was missing some load step information, as discussed in the previous section. Publications should not necessarily include all such details, but should include at least enough to allow duplication of results within acceptable engineering tolerance.
2. *Different analytical building blocks and different PBAM design style.* Local libraries of such entities could differ and, thus, cause a different PBAM master view to result. Similarly, as is also true of finite element model development, two different engineers could develop different PBAMs for the same analysis model. At present it is unclear what kind of problems this lack of uniqueness would cause, if any. Section 10.6 discusses how constraint graph theory might offer some help here.

3. *Different software tools and hardware platforms.* If different design and analysis tools are used, then it is unlikely the results will be exactly the same in the case of numerical methods-based solutions. Round-off errors and differing precision will cause differences that are beyond the control of the PBAM representation (which are hopefully negligible).

Furthermore, CAE/CAD tools can have different feature sets (related to #2 above). An example of this case occurred in the Plane Strain Model where a 10-noded element used by Lau, et al. is not in the standard ANSYS element library.

4. *Erroneous and/or Missing Datasets.* Mistakes can be present published analysis model descriptions, and some details can get left out. Related to Issue 1, this issue emphasizes the fact that PBAMs can only be as good as the information they are developed from and tested with.

10.5 Potential Benefits from Constraint Graph Theory

As mentioned briefly in Chapter 5, constraint graphs have a good theoretical foundation which is based on general graph theory. Because ABBs and PBAMs draw upon constraint graphs, this theoretical basis potentially can benefit the ABB and PBAM representations. A few possible benefits are highlighted here.

1. Constraint Solvers and Graph Algorithms

One immediate help for PBAMs with constraint-based implementations would be constraint solvers that can handle a wider variety of constraint graphs. A review of available solvers by Freeman-Benson, et al. [1990] is available. Freeman-Benson and Wilson [1990] also discuss DeltaStar, a kind of constraint solver switch that sits above flat

constraint solvers such as DeltaBlue to determine the best solver for the constraint graph at hand.

With respect to graphs in general, Cormen, et al., cover a wide variety of existing graph algorithms and note that there are "hundreds of interesting computational problems defined in terms of graphs" [1990, p. 463].

2. Computational Complexity Estimation

Related to the previous topic, constraint graph theory can be used to determine how costly a given constraint graph algorithm is in terms of run time. For example, Maloney has proven that the run time of the DeltaBlue algorithm varies linearly with the number of constraints in acyclic constraint graphs [1991]. He also has shown that graphs with cycles cause the algorithm to be NP-complete¹. Therefore, such graphs quickly require too much run time as the size of the problem increases (e.g., as the number of constraints in the graph increases).

3. PBAM Equivalence

The concept of *graph isomorphism* [Cormen, et al., 1990, p. 88] could prove useful in determining if two PBAMs developed by different people are essentially the same.

Also, such concepts possibly could help synthesize a PBAM given a set of existing ABBs and PBAMs and the extended constraint graph, G , of the desired analysis model. The "strongly connected components" algorithm described by Cormen, et al. [p. 489] might be used to identify potential subsystem subgraphs in G . After that, one could

¹ NP stands for "nondeterministic polynomial time." To date no NP-complete problem is known to have polynomial runtime [Cormen, et al., 1990, p. 927]. In other words, NP-complete problems are thought to have run times that always vary at a rate which is greater than polynomial (e.g., exponential).

potentially identify existing ABBs/PBAMs that could serve as these subsystems by checking for ABBs/PBAMs that are isomorphic with these subgraphs.

4. Automated Generation of I/O Tables

Cormen, et al. describe several graph search algorithms that can be used to determine every vertex in a graph, G , that is reachable from a specific vertex, s . For example, breadth first search [p. 469] can be used also to find the shortest path between s and each reachable vertex.

Thus, I/O tables potentially can be generated automatically by considering the desired output, O , of the ABB/PBAM to be the specified vertex, s , in the corresponding extended constraint graph. Whether or not such algorithms can categorize the reachable variables (vertices) into the appropriate I/O category (I , I' , m , a) has not been investigated.

In summary, this subsection gives at least a hint of the possible benefits that graph theory, in general, and constraint graph theory, in particular, can offer to the ABB/PBAM representation.

10.6 Implications for STEP

This section discusses how PBAMs might fit into the bigger picture addressed by STEP concerning product modeling to support the full product life cycle.

The current STEP parts for representing analysis-oriented information (e.g., Part 45 Materials and Part 103 Finite Element Analysis) are not enough to represent analysis models to the degree dictated by the Objectives in Chapter 4. In particular, no linkages between design-oriented product models and analysis models are apparent, and solution

techniques other than FEA have not been addressed yet. Also, a schema for FEA pre-processor input information has not been addressed. (Such a schema would represent information like an analytical system plus solution method parameters, symmetry considerations, etc. - information that is more or less contained in an Ansys Prep7 file). In the opinion of the author of this thesis, the reason for these gaps is that *intermediate analysis model representations are needed* (like PBAMs) to fill the gaps *due to the heterogeneous schema integration issue* identified in Chapter 2.

Therefore, a direction is suggested in Table 10.5 regarding how the ABB and PBAM representations might help STEP in regards to the above gaps. Items labeled **New** are potential libraries of analysis entities that could developed in STEP schemas by experts within the indicated field of engineering. As shown earlier in Appendix F a preliminary taxonomy of mechanical engineering analysis models has been developed in this research which can serve as a framework for developing these Parts. Finally, an application protocol in STEP that could result from application of this research is the "Exchange of Analysis Models" between CAE tools.

Another note is that *detailed product models are needed to support analysis*. The degree of detail in the product model is the limiting factor on how complex a PBAM can be for that product.

Similarly, there is an *information shortage* in current CAD/CAE tools as highlighted by the limited interface established with Mentor Graphics in this research. Much of the information needed to perform general thermomechanical analyses is not available from such PWA CAD tools, including:

- Environmental and operating conditions experienced during manufacturing and customer usage. These are generally needed to determine the loads and temperatures.
- Relations to the parent assembly. This also is needed to determine the loads and temperatures as well as boundary conditions.

Table 10.5 Potential Libraries of Analysis Entities within STEP

<u>Part</u>	<u>Title</u>
<i>STEP Integrated Resources</i>	
Generic Resources	
41	Fundamentals of Product Description & Support
45	Materials
New	Fundamentals of Product Analysis (generic ABB & PBAM structures, analytical primitives, etc.)
Application Resources	
103	Electrical Applications
104	Finite Element Analysis
105	Kinematics
New	Mechanics of Deformable Bodies (discipline-specific analytical primitives, etc.)
New	Heat Transfer, etc.
<i>STEP Application Protocols</i>	
204	Exchange of Boundary Representation Solid Models
New	Exchange of Analysis Models etc.

- PWB layup - materials, orientation, thicknesses, tolerances.
- Detailed geometry, assembly, and materials of electrical components (which often are not even in the vendor catalog).

As this information is not typically captured by present CAD tools, it must be added to the product model externally.

These latter two points indicate that the product models being developed in STEP will need to be quite detailed if they are to be used for engineering analysis purposes. One can expect that it may not be practical to all the entities to support such analysis needs. Therefore, the ability to use STEP entities to construct "standards" that are specific to the needs of a company or project may prove necessary.

10.7 Summary

This section compared the PBAM representation against the objectives for analysis model representations defined in Chapter 4. The degree that each objective is met was discussed with reference to the ABB/PBAM structure and operations defined in Chapter 5 and 6, as well as to specific examples from the case studies.

General discussion on the strengths and weaknesses of the PBAM representation was also included. Possible causes of discrepancies in PBAM representations of the same analysis model were noted, as well as in different implementations of the same PBAM. Finally, implications to the STEP project and potential benefits from constraint graph theory were described.

PART IV CLOSING REMARKS

CHAPTER 11

RECOMMENDED EXTENSIONS

*Of making many books there is no end,
And much study wearies the body.
Ecclesiastes 12:12 [NIV]*

The most obvious extensions to this research would be those identified as limitations, issues, and unfulfilled objectives (e.g., in Chapters 9 and 10). These include the following:

1. General support of global/local model interaction, including exchange of 2D and 3D boundary conditions and associated operators (Section 9.7).
2. Support for postdefined compositional topology (Section 9.7)
3. Development of general ABB operations and associated algorithms; preliminary results towards this goal were given in Chapter 5.
4. Investigation of other constraint solving algorithms for increased I/O flexibility (e.g., for handling graphs requiring simultaneous equation solution) (Sections 9.6 and 10.5).
5. Investigation of constraint graph theory to realize some of the potential benefits discussed previously (Section 10.5).
6. Development and implementation of PBAMs to represent a wider variety of analysis models. This extension can help both identify other needed constructs (as the warpage case study helped identify postdefined compositional topology) and improve PBAM Development and Implementation Guidelines (Chapters 7 and 8) by providing a broader experience base.

7. Development of general product-analysis transformations (modeling idealizations and design operations) (OBJECTIVE 3 in Chapter 10).

8. Automated selection of a PBAM and its options. (OBJECTIVE 2 in Chapter 10)

Ideally there would be some object or expert system sitting above PBAMs that is able to select the correct PBAM for a given analysis situation. Currently, the user must perform such a selection. Similarly, the flexibility afforded by supporting options in PBAMs raises the issue of how one decides which options are appropriate for a given analysis need. These two issues are related to the limitations and assumptions of the analysis model itself which are beyond the scope of the current PBAM representation.

9. Inclusion of analysis model assumptions and limitations.

This extension is felt to be a rather extensive one, but a necessary one to enable automated PBAM selection, as mentioned in the previous point.

10. Development of more complete and capable general purpose analytical building blocks. Such ABBs should be able to shift automatically between different behavior regimes (OBJECTIVE 25), and should include automatic checking of their results. Furthermore, they could be better used to wrap existing solution techniques such as more powerful automatic mesh generation for FEA.

11. Method to combine the results from several analysis models of the same problem (e.g., analysis models of varying complexity - OBJECTIVE 5) in order to produce a better final results based on all their inputs.

12. Definition of metrics to compare PBAMs of two different models (e.g., number and type of variables and relations each has).

Less research-oriented extensions would include the following:

- Graphical tools to create and use PBAMs.

One can visualize a computer-based constraint schematic / constraint graph editor / viewer (much like an electrical schematic capture tool) that would allow one to click on a subsystem and see its internal relations, swap between schematic and graph views, etc. Such a tool could prove useful in developing PBAMs and in using them (by allowing one to explore an analysis model interactively).

- Libraries of PBAMs for specific industries.

Other more long-term PBAM extensions could include the following:

- Determine Analysis "Cost" and Accuracy.
- Manage Multiple Versions of Analysis Results.
- Automate Variant Analysis.
- Automate Original Analysis.
- Include Qualitative Behavior.

CHAPTER 12

SUMMARY AND CONCLUSIONS

*Is there anything of which one can say,
"Look! This is something new"? ...
Ecclesiastes 1:10 [NIV]*

This research has produced a new representation of engineering analysis models, termed *product model-based analytical models* (PBAMs), that links product models with analysis models to enable rapid, flexible routine analysis. This structured representation automates the instantiation, execution, interaction, and, to some degree, the results feedback of a variety of routine analysis models. New characteristics of this representation include the following:

- Linkage between product models and analysis models.
- Representation of multiple analysis models of varying complexity.
- Options allowing seamless variations of analysis model behavior.
- Uniform treatment of analysis models with exact solutions (e.g., formula-based analysis models) and numerically approximate solutions (e.g., finite element solutions).

PBAMs are the first known blending of object and constraint representations for the purpose of representing engineering analysis models. Because of its foundation on these well-developed areas of artificial intelligence, the PBAM representation inherits the following characteristics:

- Modularity and flexibility.
- More intuitive representation of engineering analysis concepts.
- Declarative representation of relations.
- Multiple input/output alternatives.

Chapter 5 defined the *analytical building block (ABB) representation* and gave an initial set of general purpose ABBs that can be used potentially in many different product applications. Then the *PBAM representation* was defined in Chapter 6 as a specialization of the ABB representation which distinguishes how product information is linked with an analysis model. The following views of the ABB and PBAM representations were defined in those same chapters:

- ABB Structure / PBAM Structure
- Constraint Schematic
- Object Relationship Diagram
- Subsystems
- Extended Constraint Graphs
- I/O Tables
- Instance Views

All but the last view are *structural views* in that they capture the information that describes the structure of a particular analysis model. The ABB/PBAM structure is the master view of an ABB/PBAM from which all other structural views are derivable. The other structural views each communicate different aspects of the ABB/PBAM structure to increase human comprehension and to aide the development, implementation, and use of ABBs and PBAMs.

The last view, the instance view, is an *operational view* that shows the usage of a particular instance of a specific type of ABB or PBAM. Thus, besides the defined structure evident in the structural views, the ABB and PBAM representations also have defined operations that have been discussed in preliminary form in this thesis. Chapters 7 and 8 provide preliminary guidelines on PBAM *development* and *implementation*, respectively, that utilize the above views.

PBAMs were developed and implemented in Chapter 9 to represent case study solder joint fatigue analysis models. PBAM representations of such analysis models from the literature with both formula-based and finite element-based solutions were demonstrated. Chapter 10 evaluated the PBAM representation and how it performed in these case studies against the objectives initially set forth in Chapter 4. This evaluation showed that the PBAM representation accomplishes many of those objectives - some of which are summarized in the characteristics listed towards the beginning of this chapter.

Tough problems for the PBAM representation were also identified in Chapter 10, including representation of analysis models with postdefined compositional topology (as highlighted by the PWA warpage case study), and limited I/O alternatives (limited by the present capabilities of analysis model solution tools and general purpose constraint solvers).

Finally, recommended extensions were given, including the inclusion of analysis model assumptions and limitations into the PBAM representation, the automated selection of PBAMs and associated options, and the development of a larger set of general purpose ABBs.

Customers of Research & The PBAM Life Cycle

In the end, it is hoped that the PBAM representation eventually proves to be useful in day-to-day engineering practice. The discussion in Chapters 9 and 10, and the recommended extensions in Chapter 11 indicate that more research and development needs to occur to progress towards this end. It is with this long term vision in mind that this summary of the ultimate intended customers of this research is given. At the same time, the following summary serves as a review of how PBAMs can be developed, implemented, and used.

The first-level envisioned customers of this research are *CAD/CAE tool vendors*, who could apply these concepts to provide greater design and analysis integration to their

customers. Existing design and analysis tools should support greater interoperability to fit within CAD/E frameworks, in general, and provide seamless design and analysis using PBAMs, in particular. CAD vendors should extend their tools to capture the more detailed product information that is required by analysis. CAE vendors should add the general purpose analytical building block (ABB) layer to their products in order to provide flexibility and a more general representation of analysis models. Somewhere in between CAD and CAE is the possibility for vendors to write totally new products that enable the creation of new PBAMs, and to develop libraries of PBAMs for specific industries.

The intended second-level beneficiaries of this research are *engineering analysts* who support product design. After they have developed new analysis models and have proven them with the help of experimentalists, analysts could *develop* PBAMs that represent their new analysis models to document them precisely. This documentation would be embodied in the PBAM views defined in Chapter 6. Then, to maximize the usefulness of their analysis models, analysts could *implement* the resulting PBAMs into CAD/E frameworks (using the new vendor tools described above) and transfer them to design engineers for routine use during product design.

The final envisioned PBAM end-users are the *product designers* themselves. It is their job and the author's previous experience as a product designer that has motivated much of this research. First, they could use the PBAM documentation developed by the engineering analyst to understand an analysis model and what product data it impacts. Finally, they could use the implemented PBAMs to perform analysis with more frequency and less tedium than is now possible. This new capability for rapid, flexible analysis is intended to enable more detailed, consistent evaluation of more design alternatives. Thus, the bottom-line objectives of the PBAM representation are to design products better and to design better products.

APPENDICES

APPENDIX A

BACKGROUND MATERIAL

A.1 Overview of the Object Representation

Highlights of object-oriented concepts are given here using Smalltalk terminology as adapted from LaLonde and Pugh [1990] and Friedman [1991]. Synonyms for each concept are included in brackets. Figures A.1 and A.2 illustrate some of these concepts with a simple example using the ISO EXPRESS-G notation described in the next section.

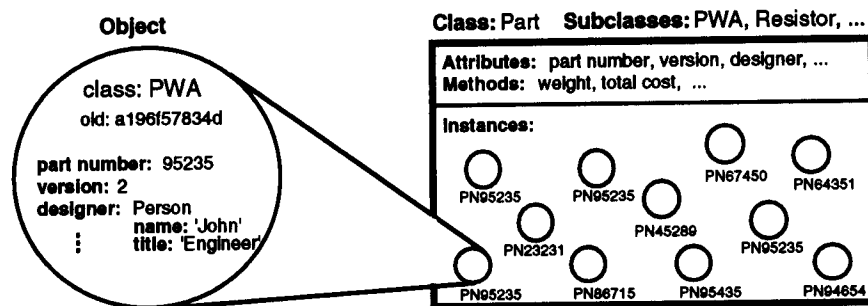


Figure A.1 Example of Object and Class Concepts

attribute [variable]: **Attributes** are the distinguishing characteristics that help define one type of object from another.

object [entity instance, instance]: An **object** is an extension of an abstract data type in that it encapsulates private data, which are the values of its attributes, but also has an associated behavior. The kind of attributes (**instance variables**) and the **methods** (functions an object can perform, i.e., its behavior) that an object has are determined by what kind of object it is, i.e., the **class** it belongs to. A **message** sent to an object must refer to a method that the object can understand. A message may require that an object

act upon itself to change its state, or that it act upon other objects by further message passing.

In the example of Figure A.1 an object is shown which is an instance of a **Part**. (the names of classes of objects are capitalized in Narrow Helvetica type). The values of its attributes can themselves be other objects (e.g., a **Person** object). An object identifier (OID) is usually automatically generated to make each instance of **Part** unique because two distinct instances could conceivably have the exact same attribute values. The method named `weight` in the class description would calculate the weight of the part based on its geometry and material(s).

class [type, entity, entity type]: A **class** is the template that defines which attributes an object has and which methods an object can understand. An object is an **instance** of a class (i.e., a unique member of the class). Objects are **instantiated** (i.e., created) by their class. Objects that are members of the same class differ only by the value of their object identifier and possibly by the local values of their instance variables (i.e., their private data). A class has a name, instance variables, class variables, and a set of methods.

class attribute [class variable]: An attribute is a **class variable** if the definition of the *class* includes a *value* for that attribute. In other words, each instance of the class uses the same value for this attribute.

methods [functions]: A method is a procedure defined as part of the class description. The methods of a class specify how its objects will behave in response to any particular message. When an object receives a message, the object takes the method header in the message and looks for the code contained in its class description to see how to respond.

messages [function calls]: The **message** is the basic control structure in object-oriented programming. It is the means by which one may communicate to an object and by which objects communicate to each other. A message is like a function call that is sent to an object. A message consists of a **receiver** (the object being sent the message), a **selector** (the method name), and optionally one or more **arguments** (other objects) to pass to the method. Below is a possible message for the example given in Figure A.1. Here `pwa1` is the receiver which is an instance of the `PWB` class, `partNumber:` is the selector, and `123456` is the argument which is a member of the `Integer` class. This message would assign the argument to the object's `partNumber` instance variable, assuming the method `partNumber:` were so defined.

```
pwa1 partNumber: 123456.
```

class hierarchy: The **class hierarchy** organizes object classes by the **is-a relationship**. "B is-a A" means that B is a special case of A (B is a **subclass** [**subtype**] of A) and that A is a general case of (A is a **superclass** [**supertype**] of B). In other terms, a superclass is a **generalization** of its subclasses, and a subclass is a **specialization** of its superclass. In Figure A.2 `Part` is a superclass, and `PWB`, `PWA`, and `Electrical Component` are its subclasses. A class passes down its instance variables and method definitions to its subclasses as defined by the class hierarchy. The root class provides the necessary structure for all classes.

inheritance: A subclass **inherits** the instance variables and methods of its superclass. If the superclass itself has superclasses, the subclass inherits the characteristics of those classes as well. A subclass may redefine any of the characteristics for its own purposes. By enabling the re-use of methods by many subclasses, high level intent and derivation can

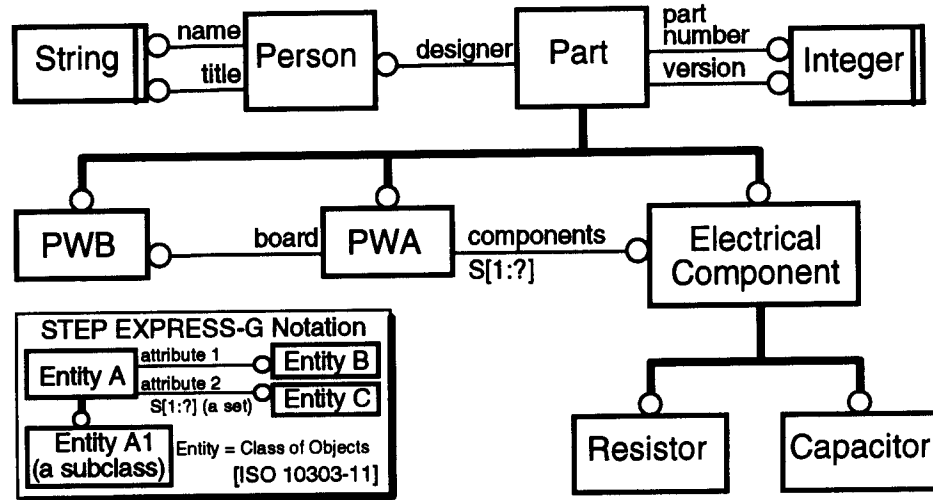


Figure A.2 Example Class Hierarchy

be captured. Inheritance also typically lessens development effort redundancy and simplifies application debugging and maintenance. **Multiple inheritance** occurs when a class is a subclass of two or more classes.

Polymorphism: When the same message is sent to different kinds of objects, each object can respond in kind because it potentially has its own specialized method as defined by its class. In traditional languages this is called operation overloading and must be implemented by extensive checking of data types. In object-oriented programming this type checking can be performed automatically as an integral function of the system. Thus, the message $c * d$ will call the method named "*" from the class that c is a member of. Therefore, the same message "*" could be used to multiply two numbers, two matrices (assuming consistent sizes), or a matrix and a number.

A.2 EXPRESS-G Notation

EXPRESS-G is a graphical notation of the information modeling language EXPRESS, which is an ISO standard [ISO 10303-11] that is one Part of STEP. The quasi object orientation of EXPRESS and its standardization make it an attractive alternative to other data modeling notations such as IDEF1x. Below is a summary of a subset of the EXPRESS-G notation used in this document (Figure A.3) which has been adapted from a draft of the EXPRESS language standard [ISO 10303-11]. A simple example is given in the preceding section (Figure A.2). The extension made by the authors to indicate class variables (using capitalized attribute names, a Smalltalk convention) is also given.

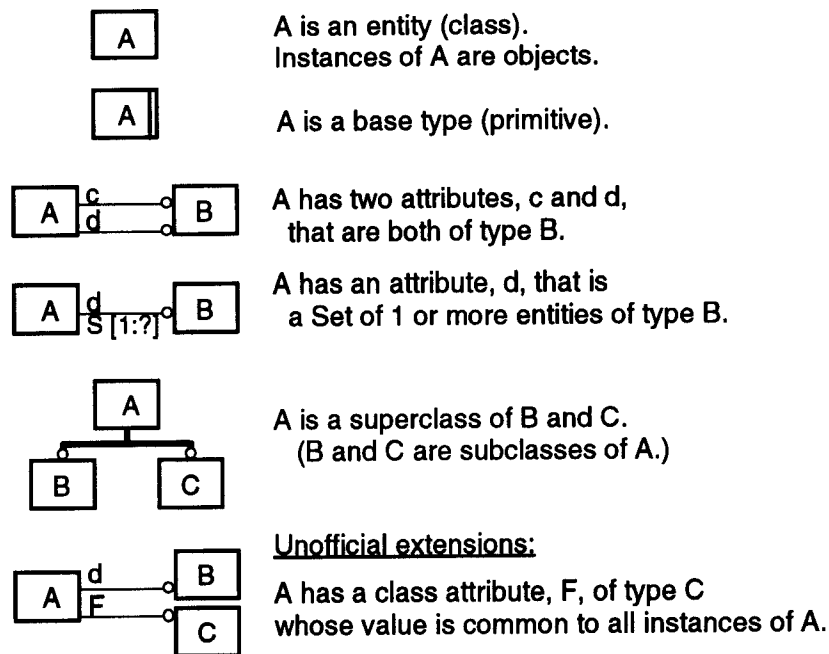


Figure A.3 Basic EXPRESS-G Notation with Extensions

A.3 IDEF₀ Notation

IDEF₀ [1981; Bravoco and Yadav, 1985] is a methodology for representing process models which has been used here to model analysis processes. Figure A.4 illustrates the basic elements of the notation with informal extensions by C-Y. Wang and the author of this thesis. The rectangular box represents a process step with information inputs on the left and information outputs on the right. Below the box is the name of the agent (which can be a human, an organization, or a computer-based tool) that performs the process step. The rounded box represents information outputs/inputs of/to the preceding/proceeding process step in a given state. The device which contains the information and the format of containment is indicated. The container might be a database and the format might be a standard format such as STEP.

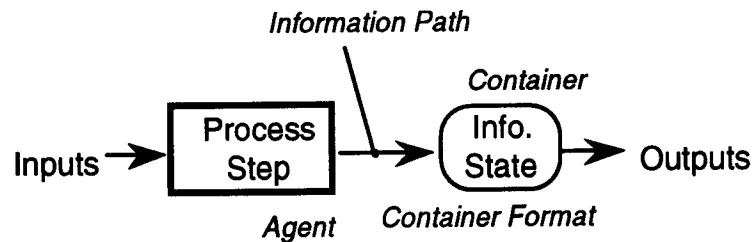


Figure A.4 Extended IDEF₀ Notation

With this extended notation, the process model can be mapped directly into an object-oriented implementation. The agent for a given box becomes an object that can perform the process step. The process step is captured as a method of the agent, and information inputs and outputs are objects. The decomposition of each process step in IDEF₀ also can be captured by having each method call upon the lower level agents in the decomposition to perform subtasks to accomplish the overall task. At the point when only one object is passed as input into a process step, the input object itself becomes the agent who calls its own method to perform that process step.

APPENDIX B

SOLDER JOINT FATIGUE ANALYSIS MODEL DESCRIPTIONS

This appendix contains two of the papers which describe the analysis models used in the case studies of this research. They have been included since such papers are one common type of analysis model description; therefore, they are representative inputs to the general development process that creates PBAMs of routine analysis models (see Chapter 7, Preliminary PBAM Development Guidelines). For the case studies performed in this thesis, the papers describe solder joint fatigue analysis models.

Engelmaier, W., 1983, "Fatigue Life of Leadless Chip Carrier Solder Joints During Power Cycling," *IEEE Trans. on Components, Hybrids, and Manufacturing Technology*, Vol CHMT-6, No. 3, Sept. 1983, pp 232-237.

Lau, J. H., Rice, D. W., Avery, P. A., 1986, "Nonlinear Analysis of Surface Mount Solder Joint Fatigue," *Proc. IEEE CHMT Intl. Electronic Mfg. Technology Symposium*, San Francisco CA, Sept. 15-17, 1986, pp 173-184.

Permission to reprint these papers have been gratefully received from the publisher.

Fatigue Life of Leadless Chip Carrier Solder Joints During Power Cycling

WERNER ENGELMAIER

Abstract—An analytical method is described which provides estimates to first order of the number of either power or environmental cycles leading to solder joint failure. Various parameter variations such as solder joint height, ceramic chip carrier (CCC) size, printed circuit substrate (PCS) material, etc. are investigated and discussed and sample estimates for a 0.65 × 0.65-in CCC are given.

INTRODUCTION

PAST EFFORTS to test the attachment reliability of leadless ceramic chip carriers (CCC's) mounted on printed circuit substrates (PCS's) have been carried out almost exclusively by temperature cycling in environmental chambers rather than by power cycling (power on/off with or without ambient temperature cycling) simulating actual operating cycles. This has led to an emphasis on matching the coefficient of linear thermal expansion α of the PCS to the coefficient of the CCC [1]–[7]. Matching thermal expansion coefficients, of course, assures meeting MIL-specification requirements which prescribe temperature cycling from -55 to $+125^\circ\text{C}$. These temperature extremes, which can cause behavior in some of the constituent materials which is nonrepresentative of normal operating conditions, might be meaningful for military applications but can lead to misleading or wrong conclusions for applications where cycling stresses are not imposed by such temperature extremes but by power cycling and/or smaller ambient temperature variations [8]–[10]. Furthermore temperature cycling can produce neither the temperature gradients, which can lead to cyclic warpage [9]–[12], nor the transient conditions [8]–[10] resulting from power cycling and which can lead to increased cyclic strains on the solder joints. Power cycling has the potential, at least in consumer applications, of being a considerably more severe reliability problem than the environmental temperature variations, because power cycling can occur more frequently. In consumer applications, large numbers of power cycles ($\sim 1000/\text{year}$) are not unlikely.

SOLDER JOINT RELIABILITY PROBLEMS

In power cycling, as in environmental temperature cycling, the reliability concern arises from the cyclic strains on the solder joints between CCC's and the substrates to which they are attached. These cyclic strains result in cumulative cyclic fatigue damage leading to solder joint fatigue failure. The

cyclic strains consist of a number of strain components: 1) strains from in-plane steady-state expansion mismatch, 2) strains from in-plane transient expansion mismatch, 3) strains from warpage due to expansion mismatch, and 4) strains from warpage due to power cycling temperature gradients. The steady-state in-plane shear strains can be readily determined analytically from the geometric and thermal design and the materials involved and is the primary strain component addressed here. The other strain components are not readily determined, either analytically or experimentally. An attempt has been made in [8] to give a worst case estimate of the transient strains for a hypothetical operating scenario. Work has recently been reported which addresses the strains on the solder joints caused by cyclic warpage [10]–[12]. In particular [12] gives an extensive discussion of the deformations and strains resulting from both environmental temperature and power cycling. While other than in-plane steady-state strains can be significant [8]–[12], indications are that the in-plane steady-state strains are of primary importance for the reliability in use conditions that involve only moderate temperature extremes. Limited temperature and power cycling results have shown that an analysis based solely on the in-plane steady-state thermal expansion mismatch is likely to underestimate the "effective strain" on the solder joints by about 25 percent.

The in-plane steady-state thermal expansion mismatch $\Delta(\alpha\Delta T)_{SS}$ is from [8]:

$$\Delta(\alpha\Delta T)_{SS} = \alpha_C(T_C - T_o) - \alpha_S(T_S - T_o) \quad (1)$$

$$= (\alpha_C - \alpha_S) \cdot (T_C - T_o) + \alpha_S(T_C - T_S),$$

where

$$\alpha_C = \frac{1}{T_C - T_o} \left(\frac{\Delta L_C}{L_C} \right) \quad \alpha_S = \frac{1}{T_S - T_o} \left(\frac{\Delta L_S}{L_S} \right)$$

α_C, α_S coefficients of linear thermal expansion for CCC and substrate, respectively,

T_C, T_S temperatures of CCC and substrate (beneath CCC), respectively,

T_o power off, steady-state temperature.

During unpowered environmental temperature variations and temperature cycling in an oven $T_C \equiv T_S$ and (1) becomes

$$\Delta(\alpha\Delta T)_{SS} = (\alpha_C - \alpha_S) \cdot (T_C - T_o) \quad T_C \equiv T_S \quad (2)$$

and $\Delta(\alpha\Delta T)_{SS}$ varies directly with $\Delta\alpha$. However during power cycling, $T_C \neq T_S$ and matching the thermal expansion coefficient of the substrate to the CCC material coefficient, i.e.,

This paper was published in the *Proceedings of the Technical Program of the 2nd Annual International Electronics Packaging Society Conference*, San Diego, CA, November 15–17, 1982.

The author is with Bell Laboratories, Whippany Road, Whippany, NJ 07981.

$\alpha_C = \alpha_C$, does not eliminate the $\alpha_S(T_C - T_S)$ term in (1). Thus even when the thermal expansion coefficients for CCC's and PCS's are matched, a thermal expansion mismatch exists during power cycling and is

$$\Delta(\alpha\Delta T)_{SS} = \alpha_S(T_C - T_S). \quad T_C \neq T_S \quad (3)$$

Unless the CCC and the substrate can be closely thermally coupled to obtain $T_S \approx T_C$, only tailoring α_S such that the thermal expansions of CCC and substrate match will eliminate $\Delta(\alpha\Delta T)_{SS}$. From (1) it follows that

$$\alpha_S = \alpha_C \frac{(T_C - T_0)}{(T_S - T_0)} \quad * \quad \text{CTE tailoring} \quad (4)$$

is necessary to eliminate $\Delta(\alpha\Delta T)_{SS}$. It has to be noted, however, that $\Delta(\alpha\Delta T)$ during the power-up and power-down transients cannot be eliminated even if the steady-state $\Delta(\alpha\Delta T)_{SS}$ is zero, since the temperature transients of CCC and substrate are different in the absence of perfect thermal coupling.

Thermal cycling is an unsuitable test for assessing the power cycle reliability of a design, because during oven cycling the term $\alpha_S(T_C - T_S)$ is forced to zero and the differences in the expansion coefficients ($\alpha_S - \alpha_C < 3 \text{ ppm/}^\circ\text{C}$) in designs with the expansion coefficients tailored for power cycling are too small to give appreciable expansion mismatches during oven cycling. Furthermore, cycling with power dissipation results not only in a temperature difference between chip carriers and substrate, but also in a temperature gradient through the substrate. These temperature differences can result in cyclic substrate warpage which cannot be simulated by temperature cycling [9]-[12]. Thus power cycling reliability has to be addressed with power cycling tests, which simulate as closely as possible actual operational conditions, in combination with analytical evaluations. Thus the effects of the various strain components are included and some degree of valid test acceleration can be obtained.

POWER CYCLE LIFE ESTIMATE METHODOLOGY

In order to obtain estimates of the number of functional power cycles a given design package can sustain before solder joint failures can be expected, it is necessary to determine the cyclic strains acting on the solder joints and to have a relative ship between the cyclic strains and cyclic lives of the solder used. A methodology to determine the shear strains from in-plane expansion mismatches has been presented in detail in [8]. It should be noted that a methodology that excludes possibly significant strain sources will yield optimistic estimates that have to be regarded as upper limits.

The primary cyclic strain component, as indicated in the previous section, results from the in-plane steady-state expansion mismatch between CCC and PCS given in (1). The information necessary for (1) is directly available from the thermal design information. Table I gives the effective steady-state temperatures for a ceramic chip carrier, 650 mils square, dissipating 1.25 W with the local ambient air at 65°C and 300 ft/min, on PCS's constructed of different materials together with the temperatures of these substrates.

TABLE I
EFFECTIVE STEADY-STATE OPERATING TEMPERATURES OF CHIP CARRIERS AND SUBSTRATES

CCC on	T_C [°C]	T_S [°C]
Epoxy/Glass PCS	96	88
Ceramic PCS	86	78

$T_0 = 20^\circ\text{C}$

TABLE II
THERMAL EXPANSION COEFFICIENTS USED FOR TABLE III

Material	α [ppm/°C]
Epoxy/Glass	15.0
Tailored Expansion	7.5
Ceramic (Substrate and CCC)	6.7

TABLE III
STEADY-STATE CHIP CARRIER-SUBSTRATE DIFFERENTIAL EXPANSION

Substrate	$\Delta(\alpha\Delta T)_{SS}$ [ppm]
Epoxy/Glass	511
Ceramic	-34
Tailored Expansion	0

From the temperatures in Table I, the coefficients of thermal expansion in Table II, a power-off temperature $T_0 = 20^\circ\text{C}$ and (1), the steady-state thermal expansion mismatches given in Table III can be calculated.

During steady-state operation, the differential expansion puts CCC's on epoxy/glass PCS's in tension, whereas a ceramic CC on a ceramic PCS is in compression. The tailored expansion PCS's have an expansion coefficient which satisfies (4) and thus during steady-state operation no expansion mismatch exists between CCC and PCS.

In order to determine the full cyclic strain history, it is necessary to consider the power-up and power-down transients. A power cycle consists of a heat-up transient from $T_0 = 20^\circ\text{C}$ to steady-state operation in about 6 min, a steady-state operation dwell of several hours, a cool-down transient to $T_0 = 20^\circ\text{C}$ taking about 30 min, and an off-period of several hours. These durations are important since for low-melting materials, stress relaxation which is time and temperature dependent is a major factor in the fatigue damage occurring during each cycle [13]-[20]. As indicated earlier, the transients are not readily determined. To illustrate the possible consequences of the transient strains, estimates of the full cyclic temperature histories for the steady-state conditions shown in Table I were made in [8] and are shown in Fig. 1.

Cyclic differential expansion histories (see Fig. 2) are obtained from the difference between the cyclic expansion histories of CCC's and PCS's which result from combining the cyclic temperature histories and the thermal expansion coefficients shown in Table II. Fig. 2 further illustrates the possibility of a significant expansion mismatch resulting during the heat-up and cool-down transients. The maximum expansion differential experienced is the sum of the amplitudes of the expansion differential extremes during a cycle.

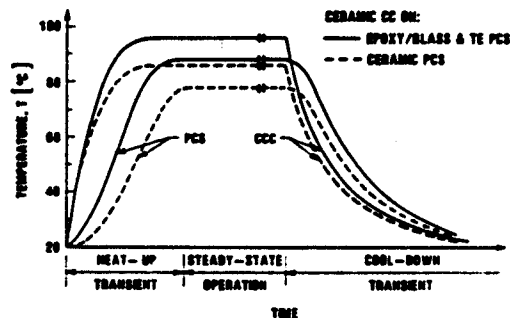


Fig. 1. Mean cyclic temperature histories of chip carriers and printed circuit substrates (CCC: 650 x 650 mil, $P = 1.25$ W, $T_{\infty} = 65^{\circ}\text{C}$ at 300 ft/min).

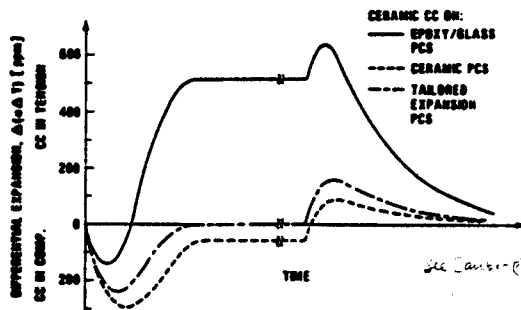


Fig. 2. Differential expansion histories of chip carriers and printed circuit substrates determined from temperature histories given in Fig. 1.

Neglecting both warpage and transient strains and assuming an ideal joint geometry giving an uniform strain distribution as well as the absence of brittle regions in the joint due to solder intermetallics or weak metallization pads on CCC or PCS, the cyclic shear strain range in a corner solder joint of height h , of a square CCC with length L , is

$$\Delta\gamma = \frac{L}{\sqrt{2}h} \Delta(\alpha\Delta T)_{SS} \times 10^{-4} \text{ in percent.} \quad (5)$$

The assumptions and conditions for (5) are somewhat related and all nonconservative. As the result of noncompliance with any of these assumptions, failures typically occur close to the bonded surfaces rather than in the center of the solder joint. Intermetallic compounds, such as Cu_6Sn_5 , Cu_3Sn , and AuSn_4 , can form rather brittle zones close to the interface areas which are more susceptible to fatigue failure [21], [22]. The joint geometry upon which the present analysis is based is a cylindrical joint with fillets at both ends which prevent both strain concentrations and offset the effects of the intermetallic embrittlement. Practical solder joint geometries tend to have larger cross-sections in the joint centers, thereby subjecting the joint regions close to the bonded surfaces to higher strains [23]. Solder joints with larger joint heights will deviate from pure shear and be subject to both tensile and compressive stresses particularly at the interfaces due to joint distortions and substrate warpage [10]-[12]. Solder joints for CCC's

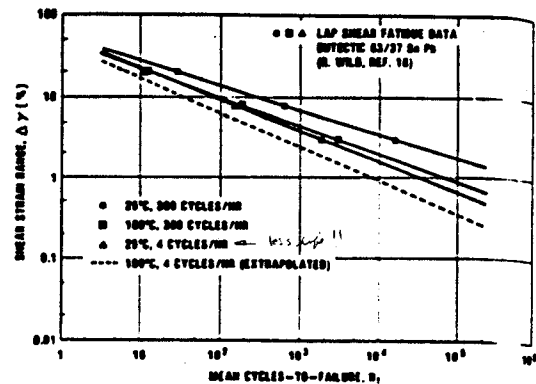


Fig. 3. Strain range-fatigue life plots for 63/37 SnPb solder at various conditions.

with castellations have much more complex joint geometries and the extent to which (5) can be used for these joints is not clear. Cyclic warpage typically occurs in the substrate, unless the substrate is stiffened, not only as a result of the differential expansion between chip carrier and substrate, but also because of the temperature gradient through the substrate.

Fatigue data for solder are scarce. The best information was developed by Wild [18], who mechanically strained solder joints at three conditions: a) 25°C and 300 cycles per hour, b) 100°C and 300 cycles per hour, and c) 25°C and four cycles per hour. The results given in Fig. 3 for eutectic tin-lead solder show that the number of mean cycles-to-failure for a given imposed strain range decreases with decreasing cyclic frequency and increasing temperature. Typical operating temperatures of the solder joints can be close to 100°C and the cyclic frequency of any on/off cycle would be four cycles per hour or less.

To derive general solder fatigue behavior for the ranges of temperatures and cyclic frequencies of interest from the available data requires the following assumptions.

- 1) The effects of the increased temperature and the decreased cyclic frequency are logarithmically superposable in the strain range direction.
- 2) Full stress relaxation takes place within 12 h and cyclic frequencies of less than one cycle per day do not further decrease cyclic life.

Some rationale for assumption 1) can be found in fundamental fatigue, which shows that fatigue effects, such as elastic strain and plastic strain, are logarithmically additive. Attempts have been made in the past [13]-[15], [17] to parametrically "correct" the Manson-Coffin fatigue-life relationship [8], [9], [24], for cyclic frequency and temperature variations. While this can provide adequate approximations for relatively high cyclic frequencies and limited variation in strain ranges, the major effect is a change in the fatigue ductility exponent c , and the change in the fatigue ductility coefficient is small (see Fig. 3).

The Manson-Coffin fatigue-life relationship is

$$\Delta\gamma = (2\epsilon_f') \cdot (2N_f)^c \quad \ln \Delta\gamma = \ln 2\epsilon_f' + c \ln (2N_f)^c \quad (6)$$

$$\bar{N}_f = \frac{1}{2} \left(\frac{\Delta\gamma}{2\epsilon_f'} \right)^{1/c}, \quad \Delta\gamma \text{ in radians} \quad (7)$$

where:

ϵ_f' fatigue ductility coefficient,
 \bar{N}_f mean cycles to failure, (i.e. $N(50\% \text{ failure})$)
 c fatigue ductility exponent.

For eutectic solder the two fatigue parameters in (7) can be determined from Fig. 3 by making assumptions about the correlations between the fatigue ductility exponent and both the solder joint temperature and the cyclic frequency. Considering the ranges of the parameters and the appropriate behavior of the exponent within these ranges, a linear temperature correlation and a logarithmic frequency correlation appear to describe this behavior best. Thus

$$2\epsilon_f' \approx 0.65 \quad \text{empirical} \quad (8)$$

and

$$c = -0.442 - 6 \times 10^{-4} \bar{T}_S + 1.74 \times 10^{-2} \ln(1 + f), \quad (9)$$

where: $\bar{T}_S = \bar{T}_{ST} + \frac{1}{4} (T_S - T_C + 2T_R)$ - ASM paper, 1989
 \bar{T}_S mean cyclic solder joint temperature, °C,
 f cyclic frequency, $1 \leq f \leq 1000$ cycles/day.

A number of authors have attempted to either establish acceleration factors to predict field failures from laboratory results [10], [15]-[17] or to directly predict fatigue life [8], [25]. The large scatter in reported fatigue results for leadless chip carriers [25], resulting from a combination of uncertainties of the thermal expansion coefficients of CCC and PCS unknown joint geometries and cyclic warpages and differing failure criteria, allows for at least some correlation of these analyses with experimental results. Most of these analyses can be improved and refined by the application of parameters derived from controlled solder fatigue testing [18]. Solder does not exhibit fatigue behavior similar to other metals. As Wild [18] has shown, the fatigue ductility coefficient c for solder does not fall into the typical range of -0.5 to -0.7 for other metals [24]. Further for the temperature and frequency ranges of interest here, the Manson-Coffin plot for solder is totally dominated by plastic deformation and does not have any significant elastic strain component as do other metals [24], [26]. The rapid stress relaxation causes elastic strains to become fully plastic deformations during the cyclic dwells. Assumption 2) requires that complete stress relaxation takes place in a 12 h time frame. While some studies indicate that as much as 50 percent of the stress relaxes almost instantaneously [16], [19], the remainder can take from minutes to hours depending on temperature [17], [19], [20]; thus 12 h appear sufficient for complete stress relaxation.

DISCUSSION

Fig. 4 shows a plot of predicted mean power cycles-to-failure, obtained from (5) and (7), as a function of solder

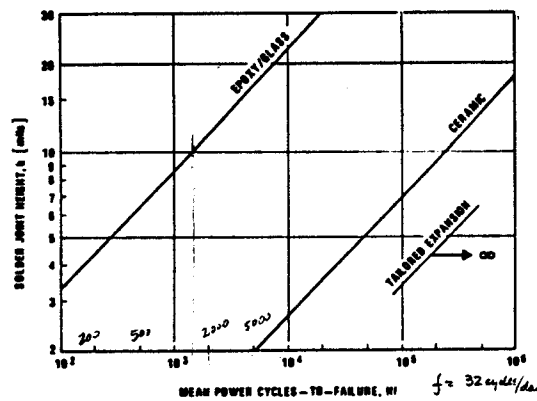


Fig. 4. Plot of number of predicted mean power cycles-to-failure as a function of solder joint height and PCS material for in-plane steady-state strains only.

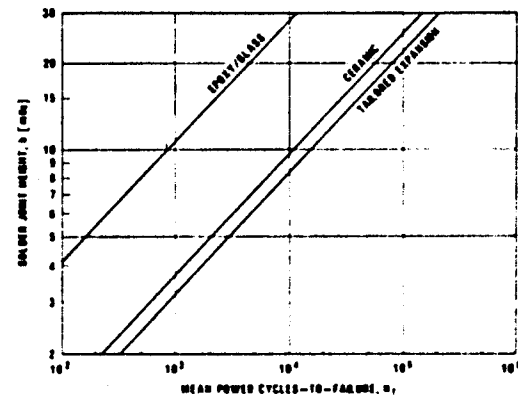


Fig. 5. Plot of number of predicted mean power cycles-to-failure as a function of solder joint height and PCS material for in-plane steady-state and transient strains. Same f

joint height and PCS material for the steady-state differential expansions given in Table III and using (5) and (7). Fig. 4 illustrates dramatically that while matching the expansion coefficients of PCS and CCC results in significant improvement in cyclic life as compared to grossly mismatched PCS/CCC materials, tailored expansion PCS/CCC combinations would not fail at all if in-plane steady-state strains were the only source for cyclic strains.

Fig. 5 shows a similar plot of predicted mean power cycles-to-failure which includes an estimate of the effects of strain components other than steady-state in-plane strains. The maximum differential expansions used to determine the shear strain range come from the curves in Fig. 2, except that only 50 percent of the transient expansion differentials were considered, since transients have a relatively short duration and are not likely to produce more than 50 percent stress relaxation [16], [19].

A comparison between Figs. 4 and 5 shows that the secondary strains caused by cyclic transients and warpage are likely to make only small contributions to the overall fatigue for

What frequency?? see notes for details 6/24/92 $\bar{T} = 32 \text{ cycles/day} = 4 \text{ cycles/hr}$ ✓

grossly mismatched PCS/CCC combinations. Conversely, for PCS/CCC combinations whose coefficients of expansion are close to being perfectly tailored for zero steady-state differential expansion, strains due to cyclic transients and warpage could be of primary importance. As can be seen from the data in Fig. 5, the choice of the substrate material can make a more than an order of magnitude difference in the functional life of a leadless chip carrier solder joint. The best substrate material is one that has a thermal expansion coefficient tailored to satisfy (4). Thus for an application requiring 1000 power on/off cycles per year, solder joint failures will occur in less than one year with epoxy/glass substrates ($h = 10$ mils), whereas the functional life on a tailored expansion substrate is about 15 years. This tailored expansion coefficient has to be somewhat larger than the CCC expansion coefficient ($\alpha_C - \alpha_{CC} \approx 0.5$ to 3.0 ppm/ $^{\circ}\text{C}$ depending on the power dissipated and the cyclic local ambient temperature change) and can be achieved for example by epoxy/glass laminates containing appropriate layers of copper-clad Invar [2], [5]. Substrates to which chip carriers with different power dissipation levels are attached can of course not be perfectly tailored for all of them if that were indeed possible for any of them. In practice it becomes increasingly more difficult to lower the thermal expansion coefficient of a laminated substrate much below 8 to 8.5 ppm/ $^{\circ}\text{C}$ and deviations from the expansion coefficient design value due to localized differences and lay-up tolerances are common. However from this study it is clear that it is not necessary to achieve perfect expansion coefficient tailoring to obtain significant improvements in power cyclic life.

Of course, substrate choice is not the only parameter that influences the mounting reliability of leadless CCC's. Other parameter choices increasing the expected functional life are 1) decreasing the ambient temperature variation during on/off, 2) increasing the solder joint height, 3) decreasing the CCC size, and 4) utilization of solder with superior fatigue properties.

It should be noted that the results in Figs. 4 and 5 imply that even at very small solder joint heights no significant thermal coupling between CC and PCS occurs. This of course is not true in reality and for small solder joint heights ($h < 5$ mils) the curves will deviate from the straight line behavior shown. It should further be noted that less than ideal joint geometries as well as brittle intermetallic compounds can lead to earlier failures and that this analysis assumes that the failure occurs in the solder joint rather than in the CCC or PCS materials or metallizations.

Some limited experimental data for both environmental temperature and power cycling have shown that cyclic life estimates from (7) are somewhat optimistic. The differences between the experimental data and the solder life predictions curve indicates that (5) underestimated the effective experimental strain ranges by 25 percent. The higher effective strains could be caused by nonideal solder joint geometries and would indicate a joint geometry factor [9], [25] of about 1.2 to 1.4; however, it is more likely that this difference is caused by a combination of nonideal joint geometries and the presence

of cyclic strains other than the steady-state in-plane shear strains accounted for in (5).

SUMMARY

An analytical method capable of predicting to first order the power cycle life expectancies of the solder joints between a leadless chip carrier and the mounting substrate is presented. This method also permits, again to first order, the correlation of fatigue data obtained at different temperatures and cyclic frequencies and the determination of acceleration factors to predict fatigue behavior in the field from accelerated laboratory tests. This method allows the assessment of the effects on these life expectancies of variations in the important design parameters. It has been demonstrated that the best power cycling performance is achieved by a tailoring of the substrate thermal expansion coefficient, rather than a matching of the substrate and chip carrier coefficients. It has also been shown that thermal cycling is not a suitable test for assessing the power cycle reliability of a design, in particular a high reliability design.

REFERENCES

- [1] R. E. Settle, Jr., "A new family of microelectronic packages for avionics," *Solid State Technol.*, June 1978, pp. 54-58.
- [2] C. L. Lassen, "Use of metal core substrates for leadless chip carrier interconnection," *Electronic Packaging and Production*, pp. 98-104, Mar. 1981.
- [3] D. Fishman and N. Cooper, "Mounting leadless chip carriers onto printed circuit cards," presented at Proc. Int. Elec. Pack. Soc., Cleveland, OH, Nov., 1981.
- [4] G. F. Love, "The impact of chip carriers on printed wiring board fabrication," *IPC Tech. Rev.*, pp. 12-19, Dec. 1981.
- [5] F. J. Dance and J. L. Wallace, "Clad metal circuit board substrates for direct mounting of ceramic chip carriers," *Electronic Packaging and Production*, pp. 228-237, 1982.
- [6] R. L. Schelhorn, "Thick-film, multilayer, chip-carrier circuit fabrication on porcelainized metal-core substrates," presented at Proc. NEPCON '82, Anaheim, California/New York, Feb./June 1982.
- [7] A. DerMarderosian et al., "A rapid technique of evaluating thermally induced strains in leadless ceramic chip carriers mounted to polymeric substrates," presented at Proc. Rel. Phys. Symp., Apr. 1982.
- [8] W. Engelmaier, "Effects of power cycling on leadless chip carrier mounting reliability and technology," presented at Proc. Int. Elec. Pack. Soc., San Diego, California, Nov. 1982.
- [9] —, "Fatigue life of leadless chip carrier solder joints," presented at NEPCON '83, Anaheim, California, Mar. 1983.
- [10] R. T. Howard et al., "A new package-related failure mechanism for leadless ceramic chip carriers solder-attached to alumina substrates," *Solid State Technol.*, pp. 115-122, Feb. 1983.
- [11] P. M. Hall, "Solder attachment of ceramic chip carrier," *Solid State Technol.*, pp. 103-107, Mar. 1983.
- [12] P. M. Hall et al., "Thermal deformation observed in leadless ceramic chip carriers surface mounted to printed wiring boards," presented at Proc. Elec. Comp. Conf., Orlando, Florida, May 1983.
- [13] G. R. Gohn and W. C. Ellis, "The fatigue test as applied to lead cable sheath," in *Proc. ASTM*, vol. 51, June 1951.
- [14] J. F. Eckel, "The influence of frequency on the repeated bending life of acid lead," in *Proc. ASTM*, vol. 51, June 1951.
- [15] K. C. Norris and A. H. Landzberg, "Reliability of controlled collapse interconnections," *IBM J. Res. Dev.*, vol. 13, no. 3, May 1969.
- [16] L. J. Merrell, "A methodology for analysis of fatigue in solder joints," Sandia Report SC-RR-71 0326, Aug. 1971.

- M. S. Rathore *et al.*, "Fatigue behavior of solders used in flip-chip technology," *J. Test and Eval.*, vol. 1, Mar. 1973.
- R. N. Wild, "Properties of some low melt fusible solder alloys," IBM Tech. Rep. No. 71Z000408, Oct. 1971.
- G. Becker, "Testing and results related to the mechanical strength of solder joints," IPC-TP-288, Apr. 1978.
- J. A. DeVore, "Fatigue resistance of solders," presented at Proc. NEPCON '82, Anaheim, California/New York, Feb./June 1982.
- "Copper-Tin Intermetallics," *Circuits Manufact.*, pp. 56-64, Sept. 1980.
- C. A. MacKay and P. E. Davis, "Solder contamination: cause and effect," presented at Proc. NEPCON '82, Anaheim, California/New York, Feb./June 1982.
- [23] L. S. Goldmann, "Geometric optimization of controlled collapse interconnections," *IBM J. Res. Dev.*, vol. 13, no. 3, May 1969.
- [24] S. S. Manson, *Thermal Stress and Low-Cycle Fatigue*. New York: McGraw-Hill, 1966.
- [25] J. K. Hagge, "Predicting fatigue life of leadless chip carriers using Manson-Coffin equations," presented at Proc. Int. Elec. Pack. Soc., San Diego, California, Nov. 1982.
- [26] W. Engelmaier, "Fatigue behavior of flex cables and circuits," *Electronic Packaging and Production*, pp. 110-118, 1979.

NONLINEAR ANALYSIS OF SURFACE MOUNT SOLDER JOINT FATIGUE

John H. Lau, Donald W. Rice and Phil A. Avery

Hewlett-Packard Company
3000 Hanover Street
Palo Alto, CA 94304

ABSTRACT

Thermal strain in surface mount chip resistor assemblies is studied by the finite element method using two and three-dimensional models. Emphasis is placed on the effects of interconnection geometry on solder joint fatigue. Nine different surface mount assemblies are considered. A simple test board with 1206 chip resistors has also been made and subjected to temperature cycling.

INTRODUCTION

Surface mounting components to printed circuit boards is one of the strongest trends in electronic packaging. Plated-through holes on the printed circuit board are used only for feed-throughs to connect the various layers of circuitry, not as the component to board interconnection.

SMT offers many advantages over conventional plated-through hole technology from many view points: cost, design, manufacturing, and quality. However, as with any new technology, SMT is not without its problems. One of the most critical issues in SMT development is the reliability of the solder joint, since the solder joint is the only mechanical means of attaching the component to the printed circuit board.

Although a variety of mechanisms (e.g., vibration, corrosion, diffusion, mechanical shock, etc.) may lead to solder joint failure, the primary mechanisms are thermal stress and low-cycle strain-controlled fatigue [1-15]. These phenomena occur when the individual parts (i.e., surface mounted component, solder joint, and printed circuit board) are subjected to cyclic temperatures. The unequal stiffness and thermal expansion of the parts give rise to stresses and strains which may damage the solder joint.

Reliability of these interconnections is generally measured by accelerated testing to failure, and by application of an analytical model to determine the life of the product under field conditions.

The most common accelerated test adopted by the electronics industry is the temperature cycling test which meets the military specification MIL-STD-883, method 1011. This method defines a thermal shock and cycling test which stresses a device by alternately switching it from a bath of hot liquid to a

bath of cold liquid, or by placing the device in a temperature cycled environmental chamber. This method requires a thermal cycle between -55 degrees Celsius and +125 degrees Celsius. Power functional cycling [2-6] would be a better approach. However, it is difficult, expensive, and time consuming [6]. In the present study only the temperature cycling approach was considered.

The most frequently used analytical model for determining the fatigue life of a solder joint is the Manson-Coffin law [16,17], Table 1. In this law, all terms are related to the physical and mechanical properties of bulk solder and joint, except the strain amplitudes which must be determined by detailed stress and strain analysis. Finite element methods will be adopted for the present analysis. The results will be presented in terms of strains. Furthermore, the whole-field deformation of the assemblies are also provided for a better understanding of solder joint fatigue.

Many values have been obtained for the fatigue life of SMT solder joints under temperature cycling. The number of cycles-to-failure varies from less than thirty to more than a thousand. However, based on the discussion in [15], this is not surprising, since SMT solder joint fatigue is a function of solder joint geometry (e.g., standoff height, fillet height and shape), pad size, paste materials, mechanical and physical properties of bulk solder and joint, surface mounted component, printed circuit board material/surface conditions, cleaning technique, testing method, failure definition/detection, cycling frequency, temperature range, dwell time at high temperature, etc. In the present study, attention is placed on the effects of solder joint geometry on SMT solder joint fatigue. Table 2 shows the cases under consideration. *

Experimental investigation of all the cases shown in Table 2 is very difficult and time consuming. The length and time scales involved made computer simulation (e.g., finite element method) particularly valuable in understanding the behavior of solder joints subjected to thermal stress and low-cycle strain-controlled fatigue. A simple test board with 1206 chip resistors has been made and tested by temperature cycling.

ANALYSIS

Figure 1 shows a surface mounted assembly. It consists of three major parts; the surface mounted component, the printed

circuit board, and the solder joint. In the present study the component is a 1206 chip resistor, the printed circuit board is FR-4 epoxy/glass, and the solder joint is a 63wt%Sn-37wt%Pb solder. Their physical and mechanical properties are shown in Table 3. It should be noted that the stiffness of the resistor is more than twenty (20) times greater than FR-4 and solder. It should also be noted that the thermal coefficients of linear expansion of the solder, resistor, and FR-4 are quite different. This suggests that in order to have a fundamental understanding of solder joint fatigue, a detailed stress/strain analysis of the whole assembly is necessary. (Figure 2) shows the complete equivalent stress-strain relation of the solder. In Figure 2, E is the Young's modulus [18], ν is the Poisson's ratio [19], α is the strain-hardening factor [20], σ_y is the yield stress [21], ϵ_y is the yield strain [22], σ is the equivalent stress [23], ϵ is the equivalent strain [24], S is the deviatoric stress tensor [25], and c is the deviatoric strain tensor [26].

It should be emphasized that the mechanical properties of the SMT materials are strongly temperature, frequency, rate, and time dependent as shown in [15]. However, for the sake of simplicity and due to the objective of this study, all the material properties are assumed to be constant. It is assumed also that the whole assembly is subjected to a temperature change of 180.0 degrees Celsius (i.e., from -55.0 to +125.0).

When a surface mounted assembly is functioning, the solder joint is subjected to a very complex state of strain due to the large thermal expansion mismatch and stiffness difference between the surface mounted component, solder joint, and printed circuit board, Table 3. Figure 3 shows a symbolic joint and the stresses acting on it. Corresponding to these stress components, there are nine strain components. Because of the highly nonlinear nature of the problem, the determination of the strains in the solder joint is very difficult. For this reason, a nonlinear finite element method will be adopted for the present analysis.

1. Finite Element Methods

It is well known that finite element methods are useful in solving boundary-value problems [27-37]. Part of the advantage stems from the ability to treat irregular boundary shapes [27] and mixed boundary conditions [30]. Due to the generality and richness of the concepts underlying these methods, they have been employed with remarkable success in solving problems in virtually all areas of engineering and mathematical physics [27-37].

The basic concept of finite element methods is that a boundary-value problem can be decomposed into a finite number of regions (elements). For each element, trial function approximations of displacement components are used in conjunction with variational principles [32] and matrix methods [33] to transform the boundary-value problem into a system of simultaneous algebraic equations. Since the method may be applied to individual discrete elements of the continuum, each element may be given distinct physical and mechanical properties, thus achieving very general descriptions of the continuum as a

whole. This feature of the finite element methods is very attractive to practicing analysts.

2. Elasto-Plastic Analysis Procedures

Due to the very low yield strength and high ductility of the 63wt%Sn-37wt%Pb solder [38-40], a large amount of plastic strain (i.e., permanent deformation) in the solder joint should be expected for each temperature cycle. The plastic strain accumulates from each cycle until the solder joint fractures. Manson [16] and Coffin [17] have developed a relation for the fatigue life. Their model states that the number of cycles-to-fracture is inversely proportional to the strain (Table 1). The effects of solder joint geometry on SMT solder joint fatigue could therefore be determined by calculating the strains in the solder joint.

In addition to plastic strain, creep strain also plays an important role in solder joint fatigue [5-6, 13-15]. In order to have a precise fatigue life prediction, an elasto-plastic-creep analysis is a must. However, for the objective of present study, it is assumed that the creep responses are the same for all the cases under consideration.

The very first step in performing an elasto-plastic analysis [41] is to select a yield surface (i.e., yield criterion) [42]. The yield surface defines how the data from the uniaxial test could be generalized through the use of an equivalent stress and an equivalent plastic strain to predict when plastic deformation would occur under a combined state of stress and strain. In the present study, the solder is assumed to obey the Distortion energy theory [43], and the equivalent stress σ and the equivalent strain ϵ are defined in Figure 2.

The next step is to define a plastic flow-rule (i.e., plastic stress-strain relation) [44]. The plastic-flow rule defines how the individual components of plastic strain depend on the stress components and the temperature histories. In the present study, the plastic strain increment of the solder is assumed to follow the Normality principle of plasticity [45].

Due to lack of supportable mechanical data of solder, the validation of the foregoing assumptions applied to solder is unknown. However, this same set of assumptions has been proved to work well for metals used in Civil and Mechanical engineering; see for example, [46-53].

The last step in performing an elasto-plastic analysis is to solve a system of simultaneous nonlinear equations [29]. Since the plastic strain depends on the state of stress and the history of loading [40], the response calculation is effectively carried out using a step by step incremental analysis [31]. For each temperature increment, equilibrium condition is achieved by an iterative scheme. In the present study, the iteration procedure is the Newton-Raphson method [33].

It should be noted that due to the large number of iterations involved in an elasto-plastic analysis, physical insight into the nature of the problem and properly

to avoid using a lot of computer time!
 defining the boundary-value problem (e.g., 2-dimensional or 3-dimensional models) are essential. In the present study, all the results are obtained by a generalized 2-dimensional analysis. The validation of the analysis is justified by the following 3-dimensional analysis.

3. 3-D Linear Analysis

Figure 4 shows a 3-dimensional finite element model for the Case 2 assembly, see Table 2. Because of double symmetries, only a quarter of the structure is modeled. It consists of 274 3-D solid elements. Each element has 20 nodal points. Each nodal point has three degrees of freedom. The linear elastic shear strain contour in the 3-dimensional solder joint is shown in Figure 5. It can be seen that the strain distribution in the Z-direction is quite uniform. This suggested that a 2-dimensional analysis of the assembly in the XY-plane is adequate. The maximum shear strain occurs near the knee of the solder joint and is equal to 0.0069.

4. 2-D Nonlinear Analysis and Results

Figures 6 through 14 present the whole-field deformations of Cases 1 through 9 assemblies, respectively. In the present study, the deformation is defined as the change in a dimension divided by the corresponding original dimension. A 2-dimensional generalized plane strain (19,26) element has been adopted for the construction of the models. This element has ten nodal points in contrast to the common eight node plane-strain element [27-33]. The extra two nodes are used to capture the deformation of the assembly in the Z-direction. Due to symmetry, only half of the structure is analysed.

For all the cases, the solder joint is subjected to the maximum deformation allowed by the solder (Figure 6-14). This is because of the extremely low yield strength/stiffness and high thermal coefficient of linear expansion of the solder. On the other hand, due to the particularly high stiffness and low thermal coefficient of linear expansion of the ceramic, the chip resistor is subjected to the minimum deformation and maximum stress.

For all the cases, Figure 6-14, due to the displacement constraints and the unequal material properties of the assembly, the maximum overall bending about the 3(Z)-axis occurs in the printed circuit board. The overall bending of the chip resistor is very small because of its small size and high flexural rigidity [19]. Nevertheless, it is interesting to note that for small standoff height (0.002"), the whole chip resistor bends in the same direction as the printed circuit board, Figure 6. The maximum deflection of the chip resistor occurs at its center. This phenomenon has also been observed by Hall [6] in his experimental work on leadless ceramic chip carriers subjected to temperature cycling. However, the overall bending of the chip resistor reduces for larger standoff heights due to the local bending and shear of the standoff solder. This behavior can be seen from the curved edges of the deformed standoff elements in Figures 7-9.

Figures 15-23 show the distributions and magnitudes of the shear strains in the isolate solder joints of Cases 1 through 9 assemblies. These shear strains are acting in the XY-plane.

For all these nine cases, the distributions of the shear strains in the solder joints are basically the same (Figures 15-23.) The maximum shear strain for all the cases occurs near the knee of the solder joint. Consequently, solder joint cracking could initiate near the corner interface of the chip resistor and the solder joint.

The magnitudes of the shear strains near the knee of the solder joints of Cases 1 through 4 (i.e., standoff heights equal to 0.002", 0.005", 0.007", and 0.01") are quite different (Figures 15-18 and Table 4.) It can be seen that the magnitudes of shear strains are larger for the assemblies with smaller standoff heights. Thus, based on Manson-Coffin law (i.e., number of cycles-to-fracture is inversely proportional to the strain), "higher" fatigue life for the assemblies with larger standoff heights should be expected. The magnitudes of the shear strains elsewhere in the solder joints are much smaller than those near the knee of the solder joints. However, it is interesting to note that the magnitudes of the shear strains at the inner end of the standoff solder are larger for the assemblies with larger standoff heights.

The effects of fillet shapes of the solder joints on the magnitudes of shear strains can be seen from Figures 16, 19-20, and Table 4 (Cases 2, 5-6). It is obvious that there is almost no effect of fillet shapes of the solder joints on the shear strains.

The effects of solder fillet height on the magnitudes of shear strains can be seen from Figures 16, 21-23, and Table 4 (Cases 2, 7-9). It can be seen that Case 8 yielded the minimum shear strain.

It is well known that for strain hardening materials, linear elastic analysis always over-predicts the stresses and under-predicts the strains. This is also true for the present study. As a matter of fact, the maximum shear strain (0.0069) of Case 2 assembly predicted by the linear elastic analysis is less than half of the value (0.0143) predicted by the elasto-plastic analysis.

5. Test Board With 1206 Chip Resistors

Figure 22 shows a simple test board with 1206 chip resistors on the left-hand side. The pad geometry is shown on the right-hand side of the board. The solder paste used was 63Sn/37Pb and its measured wet thickness was 10.5 mils.

The temperature cycling test adopted by the present study met military specification MIL-STD-883, method 1011. This method requires a thermal cycle between -55 degrees Celsius and +125 degrees Celsius. The test was stopped at 885 cycles. Up to this point there was no failure of the solder joints.

Based on the fatigue data of 60Sn/40Pb

- get this *

solder provided by Solomon [13] and the finite element calculated shear strains; it can be shown that the fatigue life of all the cases under consideration is larger than 2000 cycles.

CONCLUSIONS

An elasto-plastic analysis of 9 different surface mount assemblies has been presented in this paper. The whole-field deformations of these assemblies have also been provided for a better understanding of the behavior of solder joint fatigue. Some significant results are summarized in the following.

1. Because of the large stiffness difference and thermal expansion mismatch among the surface mounted component, solder joint, and printed circuit board, the solder joint is subjected to a very complex state of strain.
2. Due to the extremely low yield strength/stiffness and high thermal coefficient of linear expansion of the Sn/Pb solder, the solder joint is subjected to the maximum deformation.
3. Because of the particularly high stiffness and low thermal coefficient of linear expansion of the ceramic, the chip resistor is subjected to the minimum deformation.
4. Due to the displacement constraints of the solder joints and the unequal material properties of the assembly, the maximum overall bending about the 3(Z)-axis occurs in the printed circuit board.
5. The overall bending of the chip resistor is very small because of its small size and high flexural rigidity.
6. For small standoff height (0.002"), the whole chip resistor bends in the same direction as the printed circuit board.
7. However, for larger standoff heights, the overall bending of the chip resistor is reduced because of the local bending/shear of the standoff solder.
8. For all the cases under consideration, the distributions of the shear strains in the solder joints are basically the same, and the maximum shear strains occur near the knee of the solder joints. Thus, solder joint cracking should initiate near the corner interface of the chip resistor and the solder joint.
9. Based on the Manson-Coffin law, it has been shown that "higher" fatigue life should be expected for the assemblies with larger standoff heights.
10. Within the range of solder fillet shapes under consideration (standoff height=0.005"), it has been found that there is almost no effect of fillet shapes on solder joint fatigue.

11. Within the range of solder fillet height under consideration (standoff height=0.005"), it has been shown that Case 8 (solder fillet height equals to 1/3 of the height of the chip resistor) yielded the minimum shear strain.
12. Linear elastic analysis under-predicts the strains, and over-estimates the fatigue life of solder joints.
13. Because of the especially high ductility and low yield strength of the Sn/Pb solder, nonlinear plasticity analysis is a must. Better results could be achieved if creep strain is also included in the analysis.
14. The generalized two-dimensional plane strain analysis of the assemblies in the XY-plane has been shown to be adequate for the purposes of present study.
15. A simple test board with 1206 chip resistors has been made and tested by temperature cycling. No solder joint failures were observed after 885 cycles.

ACKNOWLEDGEMENTS

The authors wish to express their gratitude to Dr. Chao Chung-huei for his effective help in computer utility.

REFERENCES

1. Wild, R. N., "Some Fatigue Properties of Solders and Solder Joints," Proc. NEPCON, pp. 105-117, 1974.
2. Howard, R. T., "Packaging Reliability-How to Define and Measure It," IEEE Tr. Comp., Hybrids, Mfg. Tech., Vol. CHMT-5 (4), pp.454-462, Dec. 1982.
3. Howard, R. T., Sobeck, S. W., and Sanetra, C., "A New Package-Related Failure Mechanism for Leadless Ceramic Chip Carriers (LC-3s) Solder-Attached to Alumina Substrates," Solid State Technology, Vol. 26 (2), pp. 115-122, Feb. 1983.
4. Dixon, J. T., "Use of High I/O Chip Carriers on PWBs," IEEE Proc. 33rd Electr. Comp. Conf., pp. 500-507, 1983.
5. Engelmaier, W., "Fatigue Life of Leadless Chip Carrier Solder Joint during Power Cycling," IEEE Tr. Comp., Hybrids, Mfg. Tech., Vol. CHMT-6 (3), pp. 232-237, Sept. 1983.
6. Engelmaier, W., "Functional Cycling and Surface Mounting Attachment Reliability," Surface Mount Technology, Tech. Monograph Series 6984-02, ISHM, Silver Springs, Maryland, 1984.

7. Norris, K. C., and Landzberg, A. H., "Reliability of Controlled Collapse Interconnections," IBM J. Res. Dev., pp. 266-271, May 1969.
8. Shah, H. J., and Kelly, J. H., "Effect of Dwell Time on Thermal Cycling of the Flip-Chip Joint," Proc. ISHM Symp., pp. 3.4.1-3.4.6, 1970.
9. Hall, P. M., "Solder Attachment of Leadless Ceramic Chip Carriers," Solid State Technology, Vol. 36 (3), pp. 103-107, Mar. 1983.
10. Hall, P. M., Dudderar, T. D., and Argyle, J. E., "Thermal Deformations Observed in Leadless Ceramic Chip Carriers Surface Mounted on Printed Wiring Boards," IEEE Tr. Comp., Hybrids, Mfg. Tech., Vol. CHMT-6 (4), pp. 544-552, Dec. 1983.
11. Hall, P. M., "Forces, Moments, and Displacements during Thermal Chamber Cycling of Leadless Ceramic Chip Carriers Soldered to Printed Boards," IEEE Tr. Comp., Hybrids, Mfg. Tech., Vol. CHMT-7 (4), pp. 314-327, Dec. 1984.
12. Hagge, J. K., "Predicting Fatigue Life of Leadless Chip Carriers Using Manson-Coffin Equations," Proc. IEPS, pp. 199-208, 1982.
13. Solomon, H. D., "Low Frequency, High Temperature Low Cycle Fatigue of 60 Sn/40 Pb Solder," Symposium on Low Cycle Fatigue-Directions for the Future, Lake George, New York, October 1-4, 1985.
14. Fox, L. R., Shine, M. C., and Sofia, J. W., "Strain Rate Loading Effects in Solder Fatigue," NEPCON West, 1985.
15. Lau, J. H., and Rice, D. W., "Solder Joint Fatigue in Surface Mount Technology: State of the Art," Solid State Technology, Vol. 28, pp. 91-104, October 1985.
16. Manson, S. S., "Thermal Stress and Low Cycle Fatigue," McGraw-Hill, New York, 1966.
17. Sandor, B. I., "Fundamentals of Cyclic Stress and Strain," The University of Wisconsin Press, 1972.
18. Love, A. E. H., "A Treatise on the Mathematical Theory of Elasticity," New York, Dover publication, 1944.
19. Timoshenko, S. P., and Goodier, J. N., "Theory of Elasticity," (3rd ed.), New York, McGraw-Hill, 1970.
20. Smith, J. O., and Sidebottom, O. M., "Inelastic Behavior of Load-Carrying Members," New York, Wiley, 1965.
21. Mandelson, A., "Plasticity, Theory and Application," Malabar, Fla., Krieger, 1983.
22. Nadai, A., "Theory of Flow and Fracture of Solids," New York, McGraw-Hill, 1950.
23. Johnson, W., and Mellor, P. B., "Engineering Plasticity," Oxford, Pergamon Press, 1982.
24. Martin, J. B., "Plasticity, Fundamentals and General Results," Cambridge, MIT Press, 1975.
25. Kachanov, L. M., "Foundations of the Theory of Plasticity," Amsterdam, North-Holland, 1971.
26. Fung, Y. C., "Foundations of Solid Mechanics," Englewood Cliffs, N. J., Prentice-Hall, 1965.
27. Zienkiewicz, O. C., "The Finite Element Method," 3rd., London, McGraw-Hill, 1977.
28. Silvester, P., and Ferrari, R. L., "Finite Elements for Electrical Engineers," Cambridge, Cambridge University Press, 1983.
29. Oden, J. T., "Finite Elements of Nonlinear Continua," New York, McGraw-Hill, 1973.
30. Cook, R. D., "Concepts and Applications of Finite Element Analysis," New York, Wiley, 1981.
31. Bathe, K. J., "Finite Element Procedures in Engineering Analysis," Englewood-Cliffs, N. J., Prentice-Hall, 1982.
32. Washizu, K., "Variational Methods in Elasticity and Plasticity," Oxford, Pergamon Press, 1974.
33. Bathe, K. J., and Wilson, E. L., "Numerical Methods in Finite Element Analysis," Englewood-Cliffs, N. J., Prentice-Hall, 1976.
34. Lau, J. H., and Chao, C. H., "Finite Element Methods for Practical Engineers," to be published in Hewlett-Packard Journal.
35. Lau, J. H., and Rice, D. W., "Finite Element Modeling of Surface Mounted Assemblies," to be published in Solid State Technology.
36. Lau, J. H., and Rice, D. W., "Effects of Standoff Height on Solder Joint Fatigue," Proc. NEPCON, pp. 437-454, 1986.
37. Lau, J. H., and Rice, D. W., "Effects of Interconnection Geometry on Mechanical Responses of Surface Mount Component," to be presented in the 1986 IEEE International Electronic Manufacturing Technology Symposium.
38. Thwaites, C. J., "Soft-Soldering Handbook," International Tin Research Institute, Jan. 1982.
39. Baker, E., "Stress Relaxation in Tin-Lead Solders," Materials Science and Engineering, Vol. 38, pp. 241-247, 1979.
40. Marko, H. H., "Solders and Soldering," McGraw-Hill Book Company, 1979.
41. Hill, R., "The Mathematical Theory of Plasticity," Oxford, Clarendon Press, 1983.
42. Calladine, C. R., "Plasticity for Engineers," Chichester, Horwood, 1985.

43. Blazynski, T. Z., "Applied Elasto-Plasticity of Solids," London, Macmillan, 1983.
44. Atkins, A. G., and Mai, Y. W., "Elastic and Plastic Fractures," Chichester, Horwood, 1985.
45. Prager, W., "Introduction to Mechanics of Continua," Boston, Ginn, 1961.
46. Lau, J. H., "Bending of Circular Cylinder with Creep," Journal of Engineering Mechanics Division, Proceedings of ASCE, Vol. 107, pp. 265-270, 1981.
47. Lau, J. H., "Bending and Twisting of Pipe with Creep," International Journal of Nuclear Engineering and Design, Academic Press, Vol. 66, pp.367-374, June 1981.
48. Lau, J. H., "Creep of Pipes Under Axial Force and Bending Moment," Journal of Engineering Mechanics Division, Proceedings of ASCE, Vol. 108, pp.190-195, February 1982.
49. Lau, J. H., "Deformation of Elbows with Creep," 4th National Congress on Pressure Vessel and Piping Technology, ASME, June 1983.
50. Lau, J. H., "Creep of Thin-Walled Circular Cylinder Under Axial Force, Bending, and Twisting Moments," Journal of Engineering for Power, Transactions of ASME, Vol. 106, pp.79-83, January 1984.
51. Lau, J. H., "Creep of Thin-Walled Circular Cylinder Under Combined Loads," 5th ASCE Engineering Mechanics Conference, University of Wyoming, August 1984.
52. Lau, J. H., "Nonlinear Stress Analysis of Curved Bars," 5th ASCE Engineering Mechanics Conference, University of Wyoming, August 1984.
53. Lau, J. H., "Bending and Twisting of Pipes with Strain-Hardening," Journal of Pressure Vessel Technology, Transactions of ASME, Vol. 106, pp.188-195, May 1984.
54. Lau, J. H., "Deformation of Curved Bars with Creep," Journal of Engineering for Power, Transactions of ASME, Vol. 107, pp. 225-230, January 1985.

$$\frac{\Delta \epsilon}{2} = \frac{\Delta \epsilon_e}{2} + \frac{\Delta \epsilon_p}{2}$$

$$= \frac{\sigma}{E} (2N)^b + \epsilon_f (2N)^c$$

where $\Delta \epsilon/2$ = total strain amplitude
 $\Delta \epsilon_e/2$ = elastic strain amplitude = $\Delta \sigma/2E = \sigma_e/E$
 $\Delta \epsilon_p/2$ = plastic strain amplitude = $\Delta \epsilon/2 - \Delta \epsilon_e/2$
 ϵ_f = fatigue ductility coefficient
 c = fatigue ductility exponent
 σ_f = fatigue strength coefficient
 b = fatigue strength exponent
 E = modulus of elasticity
 $\Delta \sigma/2 = \sigma_e$ = stress amplitude
 N = cycle-to-failure

$$V = A_w = A(0.062) = 2.728 \times 10^{-5} \text{ in}^3$$

Table 1 Generalized Manson-Coffin fatigue eq.

length
base = 0.040 - Fig 16
width = 0.06

Case	Fillet height (in)	Fillet Shape	Standoff Height 10^{-3} (in)	Solder Volume 10^{-5} in^3
1	.024	concave	2	1.73
2	.024	concave	5	2.47
3	.024	concave	7	2.97
4	.024	concave	10	3.71
5	.024	straight	5	2.73
6	.024	convex	5	3.06
7	.016	concave	5	2.08
8	.008	concave	5	1.84
9	0	---	5	1.00

Table 2 Solder volume for various solder joint geometries

Nickel solder + FR-4 are pretty well matched

	YOUNG'S MODULUS (PSI)	POISSON'S RATIO	THERMAL COEFFICIENT OF LINEAR EXPANSION (in/in-°C)
SOLDER	1500000.0 1.5e6	0.40	0.000021 21e-6
RESISTOR (Aluminum)	37000000.0 3.7e6	0.30	0.000008 8e-6
FR-4	1600000.0 1.6e6	0.28	0.000015 15e-6

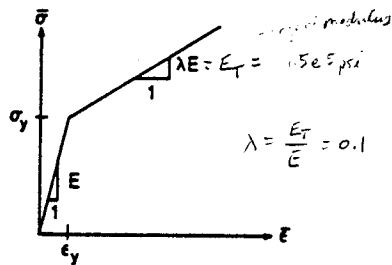
Table 3 Physical and mechanical properties of the assembly

$$\Delta T = 180^\circ \text{C all cases}$$

Case	Fillet Height (in)	Fillet Shape	Standoff Height 10^{-3} (in)	Shear Strain γ_{xy}
1	.024	concave	2	.0188
2	.024	concave	5	.0143
3	.024	concave	7	.0137
4	.024	concave	10	.0125
5	.024	straight	5	.0142
6	.024	convex	5	.0142
7	.016	concave	5	.0136
8	.008	concave	5	.0128
9	0	---	5	.0155

Table 4 Max. shear strain for various solder joint geometries

Shear Modulus, $G = \frac{E}{2(1+\nu)} = \frac{1.5e6}{2(1+0.4)} = 535,714 \text{ psi} = 0.535e6$
 $\approx 36\% E$



$E = 1500 \text{ ksi} = 1.5e6 \text{ psi}$
 $\nu = 0.4$
 $\lambda = 0.1$
 $\sigma_y = 5 \text{ ksi} = 5000 \text{ psi}$
 $\epsilon_y = 0.0033 \text{ (fracture)}$
 $\sigma = \sqrt{\frac{1}{2}(\sigma_x^2 + \sigma_y^2 + \sigma_z^2)}$
 $\epsilon = \sqrt{\frac{1}{2}(\epsilon_x^2 + \epsilon_y^2 + \epsilon_z^2)}$
 $\sigma_y = E \epsilon_y = 4950 \text{ psi} \approx 5 \text{ Kpsi}$

Figure 2 Equivalent stress-strain relation for a Sn/Pb solder

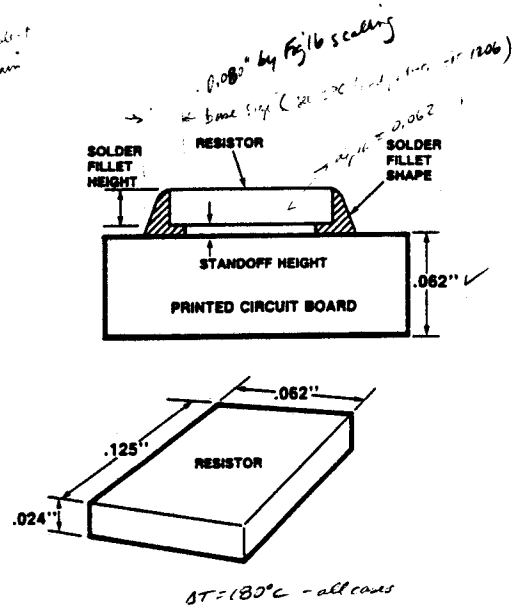


Figure 1 Surface mount assembly with a 1206 chip resistor

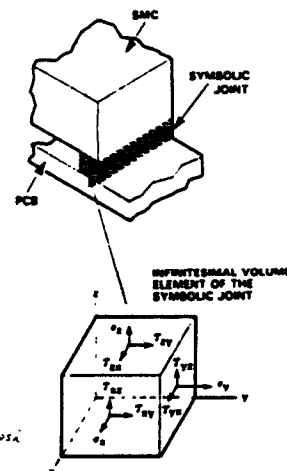


Figure 3 Stresses at a point of a solder joint

For shear yield failure: $\sigma_{max} \geq \frac{\sigma_y}{\sqrt{3}} = 0.00192501$

this page has
distorted pictures? X longer than Y (compare vs Fig. 4)

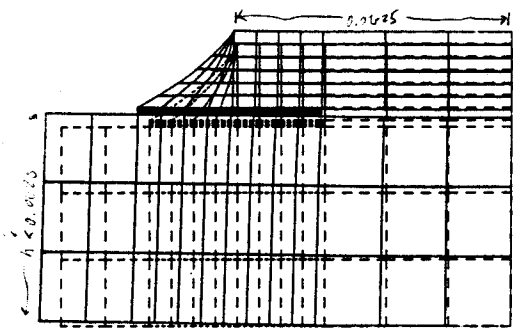


Figure 6 Deformation of Case 1 assembly

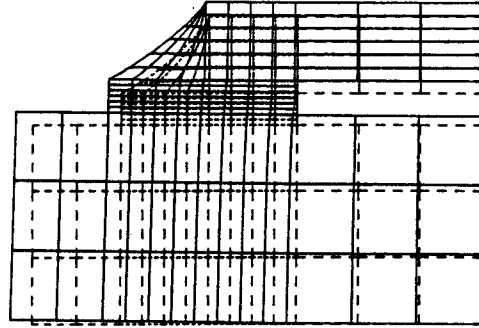


Figure 9 Deformation of Case 4 assembly

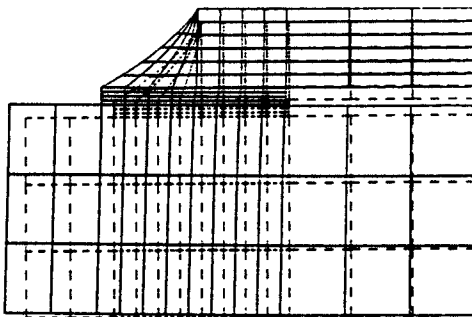


Figure 7 Deformation of Case 2 assembly

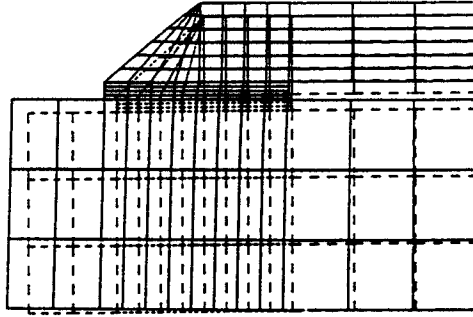


Figure 10 Deformation of Case 5 assembly

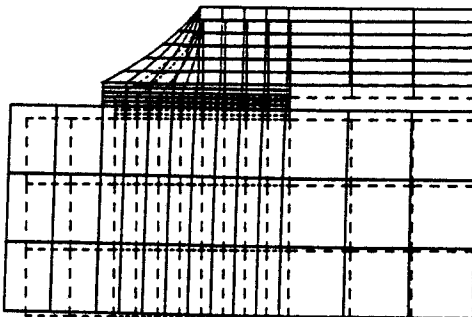


Figure 8 Deformation of Case 3 assembly

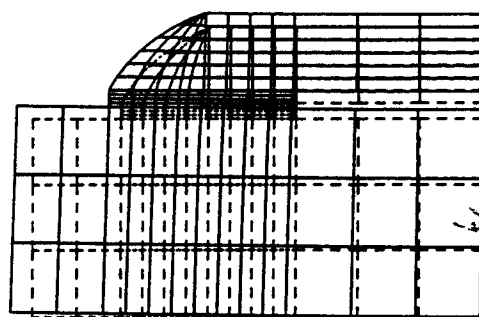


Figure 11 Deformation of Case 6 assembly

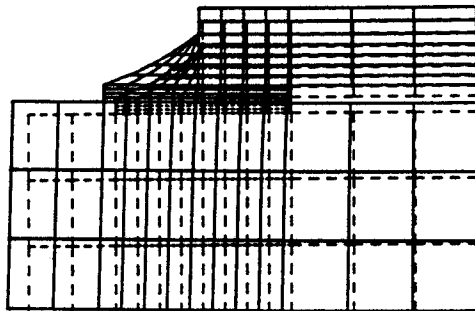


Figure 12 Deformation of Case 7 assembly

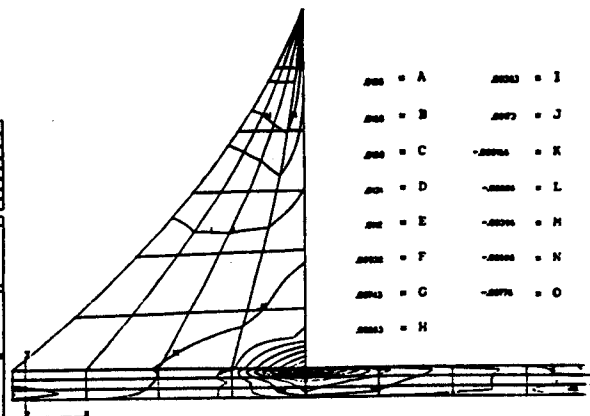


Figure 15 Shear strain in Case 1 solder joint

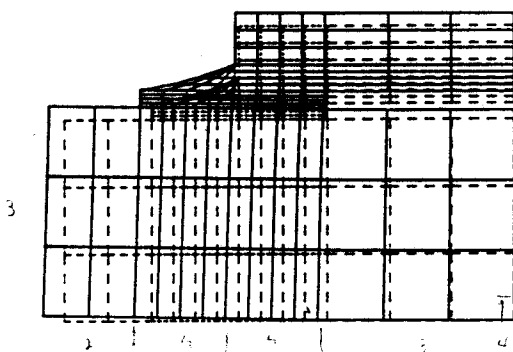


Figure 13 Deformation of Case 8 assembly

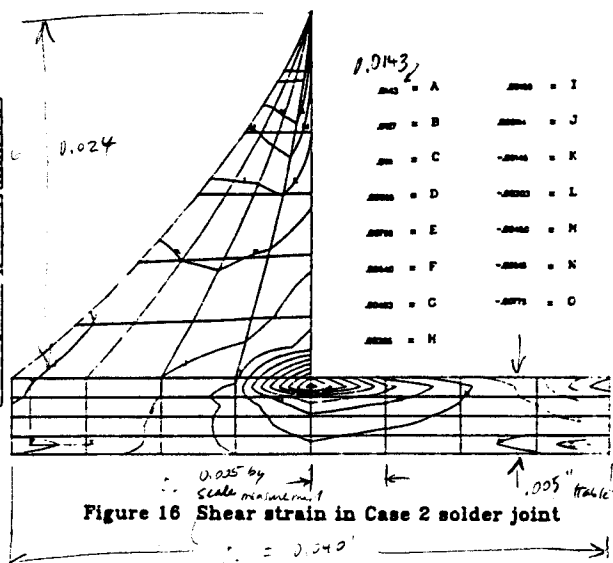


Figure 16 Shear strain in Case 2 solder joint

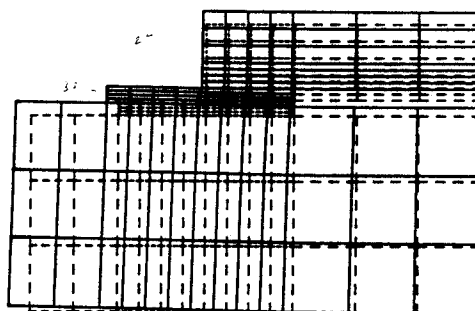


Figure 14 Deformation of Case 9 assembly

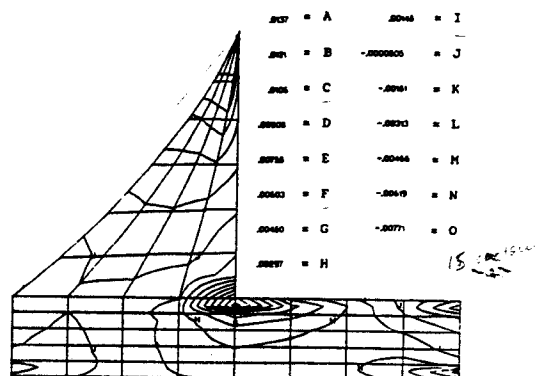


Figure 17 Shear strain in Case 3 solder joint

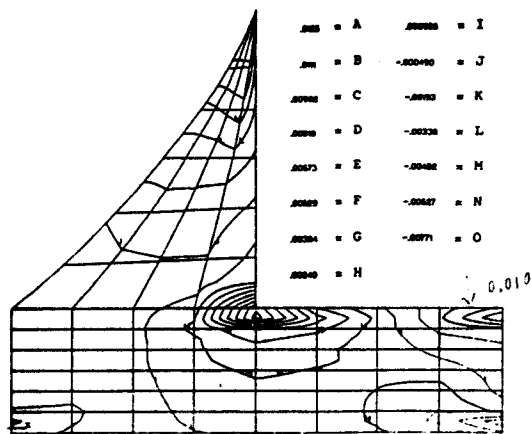


Figure 18 Shear strain in Case 4 solder joint

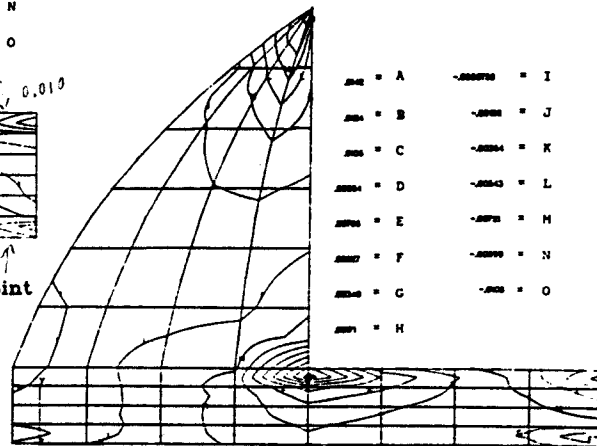


Figure 20 Shear strain in Case 6 solder joint

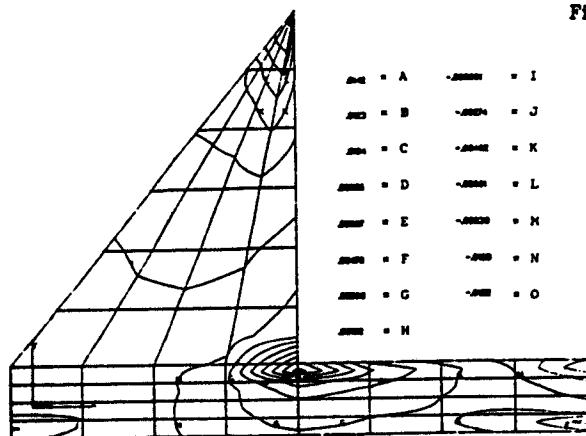


Figure 19 Shear strain in Case 5 solder joint

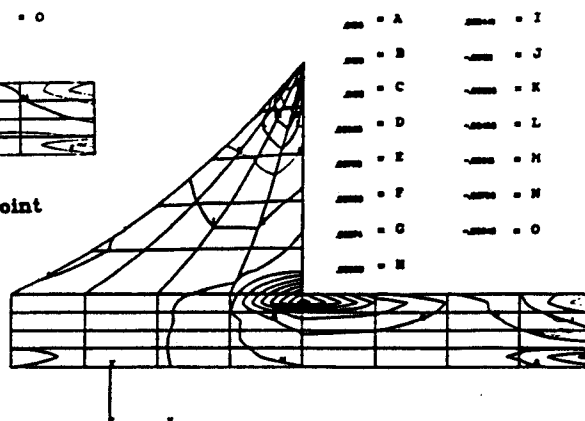


Figure 21 Shear strain in Case 7 solder joint

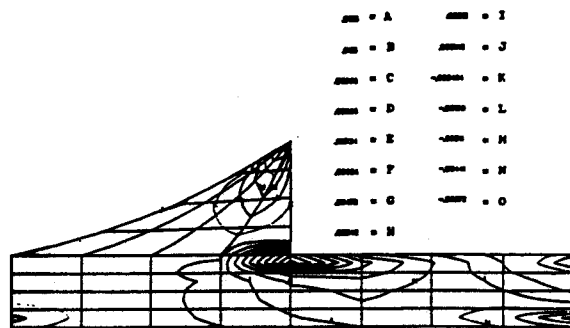


Figure 22 Shear strain in Case 8 solder joint

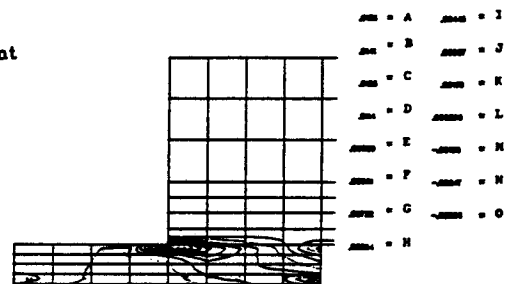


Figure 23 Shear strain in Case 9 solder joint

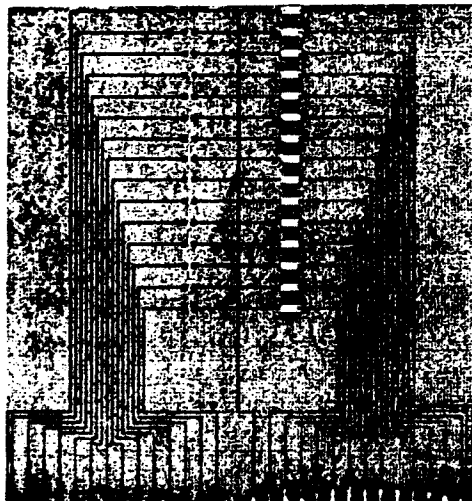


Figure 24 Test board with 1206 chip resistors


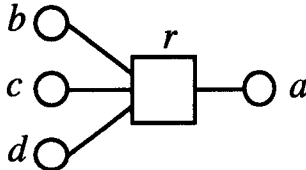

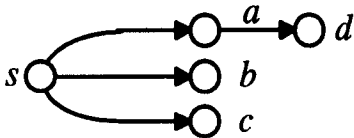
APPENDIX C

CONSTRAINT NOTATIONS

C.1 Extended Constraint Graph Notation

This section summarizes the extended constraint graph notation introduced in Chapter 5. New developments in this research that are shown in this section include new graphical symbols for an existing concept (equality relation, G3) and for a semantic extension of an existing notation (part-of relation, G4). The part-of relation (G4) is based on the data decomposition symbol by Rumbaugh, et al [1991, p. 126] who used it without constraint graph considerations. The variable symbol (G1) and the relation symbol (G2) are taken from the work of Maloney [1991, p. 33].

Extended Constraint Graph Notation

	Graphical Symbol	Meaning	Equivalent Text
G1	Variable 	a is a variable .	a
G2	Relation 	Variables a , b , c , and d are related by relation r . This relation can be written as $r(a,b,c,d)$.	$r(a,b,c,d)$
G3	Equality Relation 	Variables a and b are equal, i.e., an equality relation exists.	$a=b$
G4	Part-of Relation 	Variable s has attributes a , b , and c which are variables (i.e., they are part-of s). Variable d is an attribute of a and a subattribute or subvariable of s . I.e., variables s , $s.a$, $s.b$, $s.c$, and $s.a.d$ are shown.	Dot Form $s.a.d$ $s.b$ $s.c$ Indented Form s $\quad a$ $\quad\quad d$ $\quad b$ $\quad c$

C.2 Constraint Schematic Notation

This appendix section summarizes the constraint schematic notation introduced in Chapters 5 and 6. For familiarity purposes, a modest attempt has been made to use similar graphical symbols from electrical schematics that are analogous to the constraint graph concept being represented.

Symbols marked with an asterisk (*) have the same meaning as corresponding symbols in the extended constraint graph notation summarized above. The variable symbols (S1 and G1) are exactly the same, as are the equality relation symbols (S3 and G2); they are repeated here for convenience. For the sake of convention and style, the graphics of the relation symbols (G2 and S2) and the part-of relation symbols (G4 and S5) differ slightly. Every connection between variables and relations in a constraint graph is a separate curve per mathematical convention. Connections in a constraint schematic run vertically and horizontally and can contain equality junctions (S4) per electrical schematic convention.


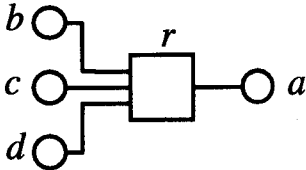

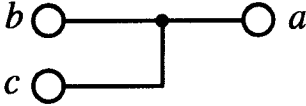
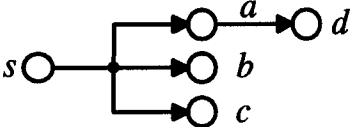
All the graphical symbols in this section are results of this research (except as noted in the preceding section). The subsystem symbol (S6) is based on integrated circuit pinout diagrams and is, thus, one realization of the "software IC" concept. This symbol merges constraint and object concepts (for relations, information hiding, and encapsulation). The extended switch concepts and associated symbols (S10, S11, and S12) are believed to be new to this research, though Leler briefly mentions simple switches as a type of higher order constraint [1988, p. 136]. Likewise, the usage-oriented instance notation (S13 and S14) are original to this research.

As some relations occur quite frequently in engineering analysis, it is convenient to introduce specialized symbols for these relations. The symbol used for the absolute value relation (M1) is based on the electrical schematic notation for a diode because they

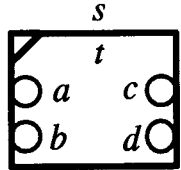
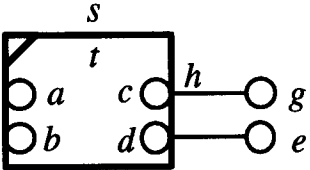
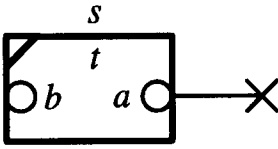
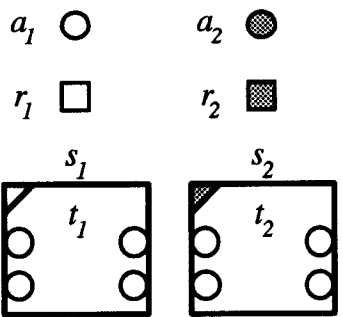
perform analogous functions in one direction: they do not let negative values pass through. The symbol used for the scale & offset relation (M2) comes from operational amplifiers that scale electrical signals by a specified gain.

Though a few such symbols are given here, it is obvious that many more could be developed. However, it is not desirable to have too many symbols or the ease of recognition intended by the graphical notation will be diminished.

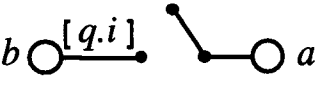
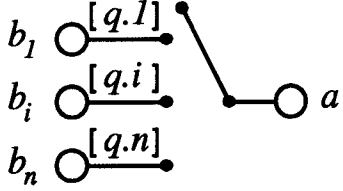
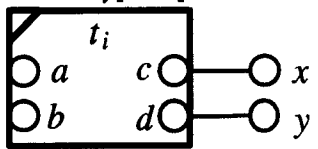
Basic Constraint Schematic Notation

	Graphical Symbol	Meaning	Equivalent Text
*S1	Variable 	a is a variable .	a
*S2	Relation 	Variables a , b , c , and d are related by relation r . This relation can be written as $r(a,b,c,d)$.	$r(a,b,c,d)$
*S3	Equality Relation 	Variables a and b are equal, i.e., an equality relation exists between them. (See Note 1)	$a=b$
S4	Equality Junction 	Variables a , b , and c are equal.	$a=b=c$
*S5	Part-of Relation 	Variable s has attributes a , b , and c which are variables (i.e., they are <i>part-of</i> s). Variable d is an attribute of a and a subattribute or subvariable of s . I.e., variables s , $s.a$, $s.b$, $s.c$, and $s.a.d$ are shown.	Dot Form $s.a.d$ $s.b$ $s.c$ Indented Form s $\quad a$ $\quad\quad d$ $\quad b$ $\quad c$


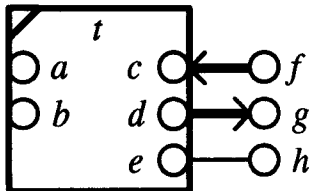
Basic Constraint Schematic Subsystem Notation

	Graphical Symbol	Meaning	Equivalent Text
S6	<p>Subsystem</p> 	Variable s is a subsystem of type t which has attributes and/or sub-attributes a through d .	
S7	<p>Semantic Linkage</p> 	Variable $s.c$ is known as h in the scope outside of t by a semantic linkage (i.e., $h=s.c$ and $g=h$). Variables a , b , and d are semantically linked with the same names in both scopes (i.e., $a=s.a$, $b=s.b$, $d=s.d$, and $d=e$).	$a=s.a$ $b=s.b$ $h=s.c$ $d=s.d$ $g=h$ $d=e$
S8	<p>Invalid Variable</p> 	Variable a is not valid in the scope outside of t .	a is <i>invalid</i>
S9	<p>Inheritance</p> 	<p>Unshaded variable a_1, relation r_1 and subsystem s_1 are inherited from the super class.</p> <p>Shaded variable a_2, relation r_2, and subsystem s_2 and related connections are new to the class containing this constraint schematic.</p>	


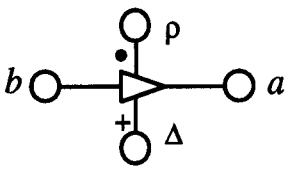
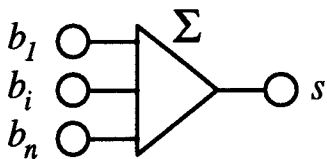
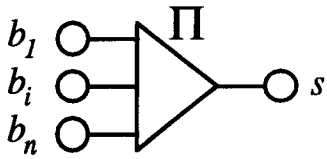
Constraint Schematic Switch/Option Notation

S10	Two Position Switch 	A two-position pole equates a and b when switch position i of switch q is selected ($i=1,2$). In other words, $a=b$ when option i of option category q is chosen.	$a=b [q.i]$
S11	Multi-Position Switch 	An n-position pole equates a and b_i when position i of switch q is selected (i.e., when option i of option category q is chosen), for $i=1...n$.	$a=b_1 [q.1]$ $a=b_i [q.i]$ $a=b_n [q.n]$
S12	Subsystem Substitution 	Switch m contains an n -position pole between each connected variable pair. (e.g., $s_i.c=x$ when position i is selected for $i=1...n$) In an ABB, option category m indicates subsystem s can be one of n possible types of objects, $s_i=t_i$, for $i=1...n$.	$s_i.c=x [m.i]$ $s_i.d=y [m.i]$ for $i=1...n$

Constraint Schematic Instance View Notation

S13	Jumper 	The analysis context has connected a and b together using a jumper .	$a \pm b$
S14	Instance View 	An <i>instance</i> of subsystem s has variable f input into variable $s.c$. Variable g is read as an output from $s.d$. In contrast, variables h and $s.e$ are always equal.	

Mathematical Constraint Schematic Notation

	Graphical Symbol	Meaning	Equivalent Text
M1	Absolute Value 	An absolute value relation exists between a and b .	$a = b $ $b = \pm a$
M2	Scale & Offset 	A scale & offset relation exists between a and b . If not shown, the value of ρ (the scale ratio) defaults to 1, and Δ (the offset delta) defaults to 0.	$a = \rho \cdot b + \Delta$, $b = (a - \Delta) / \rho$
M3	Summation 	A summation relation exists between s and b_i .	$s = \sum_{i=1}^n b_i$
M4	Product 	A product relation exists between s and b_i .	$s = \prod_{i=1}^n b_i$

APPENDIX D

PROTOTYPE CAD/CAE FRAMEWORK

The concepts developed in this research have been tested through a specific computer implementation in the Objectworks\Smalltalk object-oriented development environment on a SUN SPARCstation2. Objectworks\ Smalltalk [ParcPlace, 1991] was chosen due to its excellent prototyping environment, its extensive built-in class library, and its ready availability at Georgia Tech. It does not support automatic multiple inheritance, and its primary practical limitation is that applications cannot be run apart from the Smalltalk environment. A discussion of the engineering information system follows and the supporting product model is described in Appendix E.

D.1 General Architecture

To deal with the problem of accessing the information needed to perform an analysis, an executive-centered information integration architecture has been partially implemented as shown in Figure D.1. This prototype object-oriented engineering information system (EIS - also known as a CAD framework or information integration framework) is based on the relational implementation by Yeh [1990, 1991, 1992] and other frameworks including CFI [1991] and EIS [1989]. Extensive work has been done by Stephens [1993] to produce a kind of CAD/CAE framework (known as LEGEND) for the purpose of exploring design strategies. He includes techniques for wrapping design and analysis agents (CAD/CAE tools) and defines accumulators for storing design changes. Consequently, extensions to

this thesis might consider his work and that of the others mentioned to produce more advanced tool interfaces.

In Figure D.1, the executive, DIMS, acts as a software-based switch that connects the user with the design tools to be used (e.g., Mentor Graphics LAYOUT tool) and the data the tools require which is stored in a logically common database. The structure of the objects in this database could be based on STEP draft standards [ISO 10303-1, etc.] as described in Appendix E.

With this EIS as the test bed, sample PWAs can be built using the Mentor Graphics ECAD tools. Populated PWA objects are contained in the common database for linkage with the analytical models. Analytical primitives, analytical systems, and PBAMs

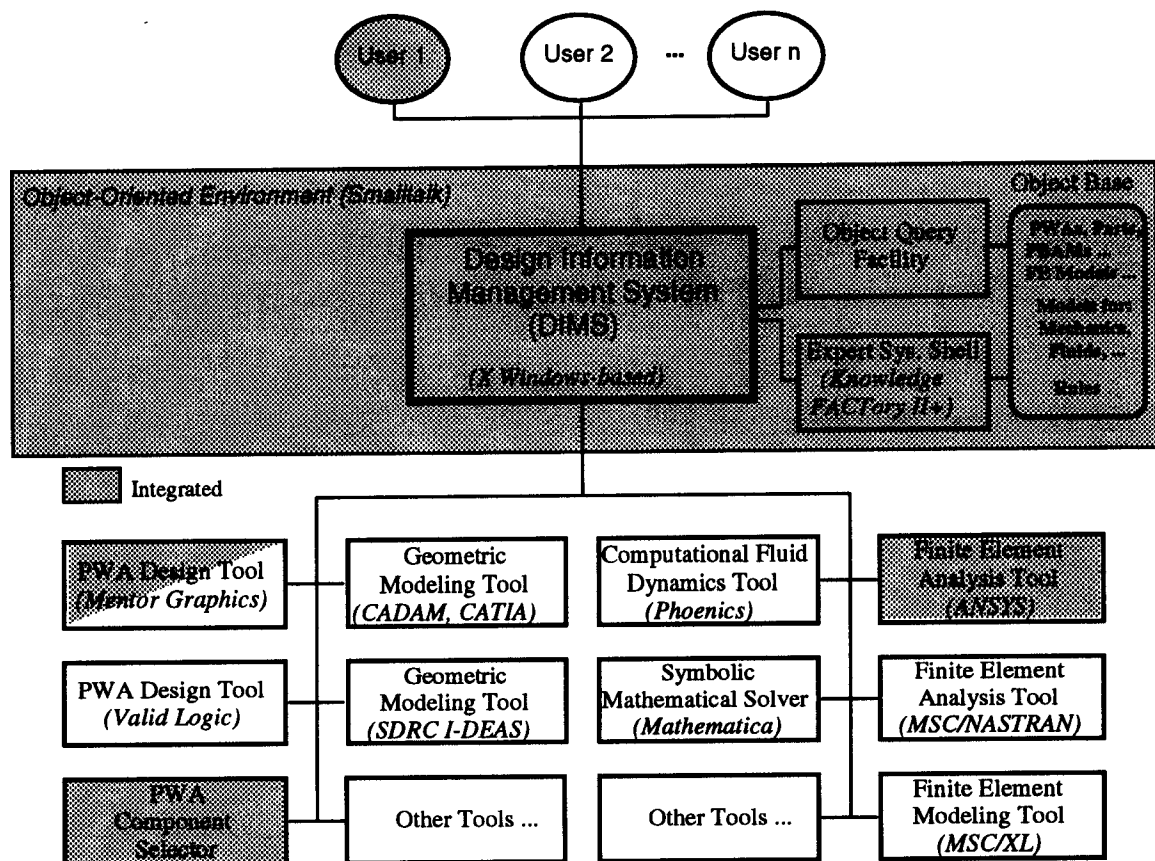


Figure D.1 Object-Oriented Engineering Information System

all are implemented as objects and are part of the database. Future extensions could include solving these analytical representations symbolically using existing tools, such as *Mathematica* [Wolfram, 1991]. The PWA component selection tool referenced in the figure, which was developed by the author and is described in a previous paper [Peak and Fulton, 1992a], demonstrates how other design tasks can be concurrently supported by this type of framework. That application is presently the only user of the prototype Knowledge FACTory II+ [Anderson, 1989] rule-based expert system shell.

D.2 PWA Product Modeling Tools

In the current implementation, a proof-of-concept linkage between the Mentor Graphics LAYOUT tool and the common database is one-way and is achieved by exporting a neutral file from LAYOUT and parsing the file to create PWA objects in Smalltalk (see Section D.4 for limitations of this link). The Smalltalk-based T-Gen tool [Graver, 1992], which was used to create the translator, provides a general approach that can be used to build links with other tools in the framework. More powerful procedural interfaces that enable dynamic interchange of information could also have been used but were not due to the focus of this research.

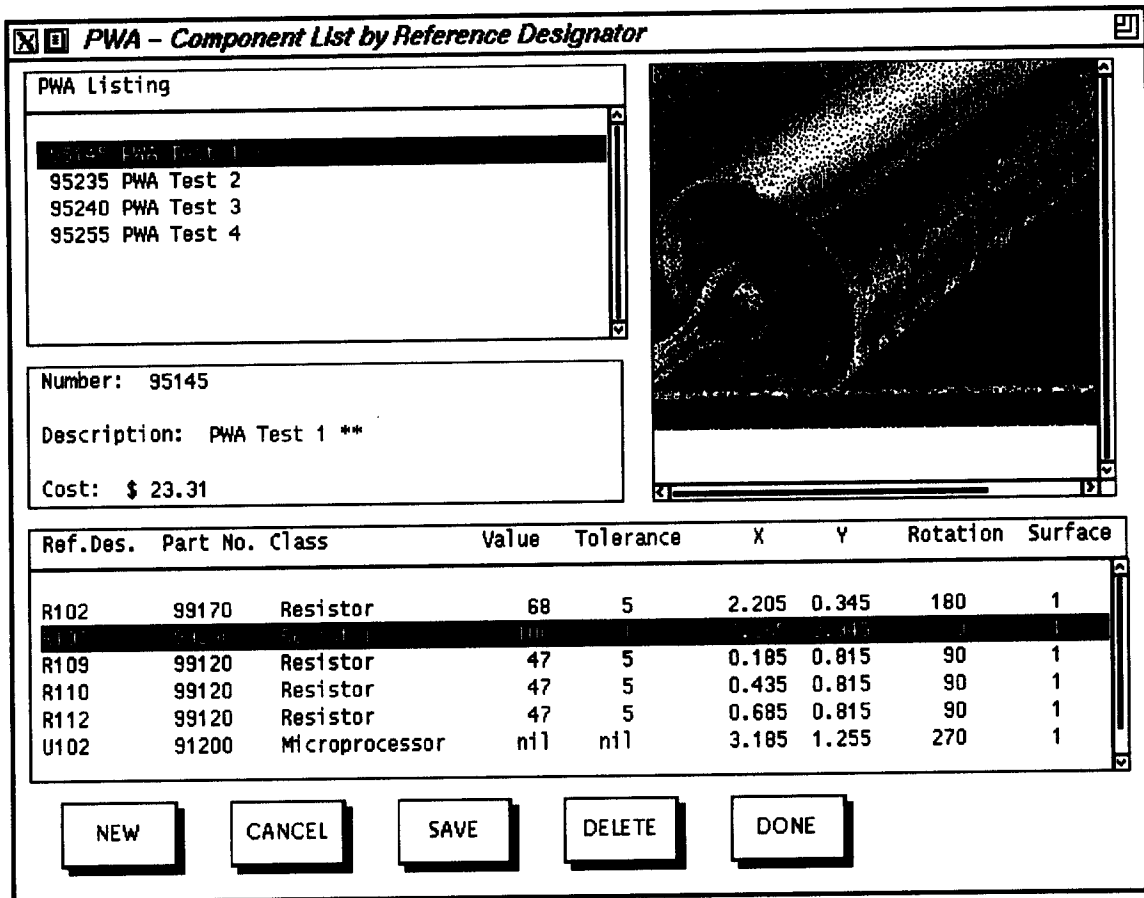


Figure D.2 View of PWA Information in the Objectbase

Presently PWA component layout information like that illustrated in Figure D.2 could be exchanged via a neutral file approach. The inclusion of other information within the component object, such as the part photo shown, is supported by the flexible object-oriented representation. FACETS, a user interface class library for Smalltalk, was used to generate this view of a representative PWA [Reasonable Solutions, 1991].

D.3 Finite Element Analysis Tool

A limited link with ANSYS was created which automatically transfers, executes, and retrieves necessary files. Chapter 9 describes the steps involved to run the case study PBAMs.

D.4 Constraint Solver

An extended version of ThingLabII was received from John Maloney and utilized in this prototype. The DeltaBlue [Freeman-Benson, Maloney, Borning, 1990] algorithm implemented in it provides the generic constraint solving capabilities described in Chapter 8.

D.5 Limitations of Prototype Implementation

In order to concentrate on the main research focus, the following limitations were allowed in the prototype:

1. A true database management system (DBMS) which would provide robustness features is not being used in the prototype; however, it is felt that the object management capabilities inherent within the Smalltalk environment are sufficient for the purposes of this research.
2. The link with the Mentor Graphics (MG) board layout software was done on a "proof-of-concept" basis. The structure of the neutral file MG exports/imports was used to develop a parser (with the T-gen tool [Graver, 1992] which could read in PWA data like that shown in Figure D.2. Using a neutral file exported from an example PWA that came with the tool, the parser created actual PWB/PWB/Component objects in Smalltalk based on the product model in the Appendix E. However, from this exercise

it became apparent that most of the product data needed for the solder joint fatigue case studies was not captured in Mentor Graphics, so no further effort was spent in creating PWAs and components in Mentor Graphics. Instead, the PWAs and components needed for the case studies were created directly as objects in the CAD framework using the PWA product model in Appendix E.

Though such limitations would not be acceptable in an industrial setting, they are believed to be reasonable with respect to this research as they relate to aspects that are not directly the focus of this thesis.

APPENDIX E

PWA PRODUCT MODEL

An engineering analysis typically utilizes a large amount of detailed product information in the development of an analytical model. Originally in this research the author developed an entity-relationship (E-R) [Chen, 1979] data model of PWA design and manufacturing information [Fulton, Ume, et al., 1990]. This model was implemented in a relational database management system (ORACLE) and representative design-oriented queries were demonstrated. However, some observed short-comings of the relational representation are that it:

- Requires tabular data structure.
- Is limited to data integration - (it included no behavior).
- Lacks flexibility and modularity.
- Represents complex entities awkwardly.

Therefore, object-oriented product models which offer solutions to the above problems have been used. A Smalltalk implementation of objects based on STEP draft standards [ISO 10303-41, -42, -103] for the representation of PWAs was partially completed. However, the layered electrical product (LEP) schema of the ISO 10303-103 draft has been determined to be insufficient for general usage. Therefore, an alternate Federated Model being developed at PDES, Inc., which is attempting to harmonize many PWA data models (including VHDL, EDIF, IGES, IPC, LEP, and RAMP), could be considered for research extension usage when available [PDES, 1991]. Since none of the PWA product data models listed currently include sufficient information to support thermomechanical

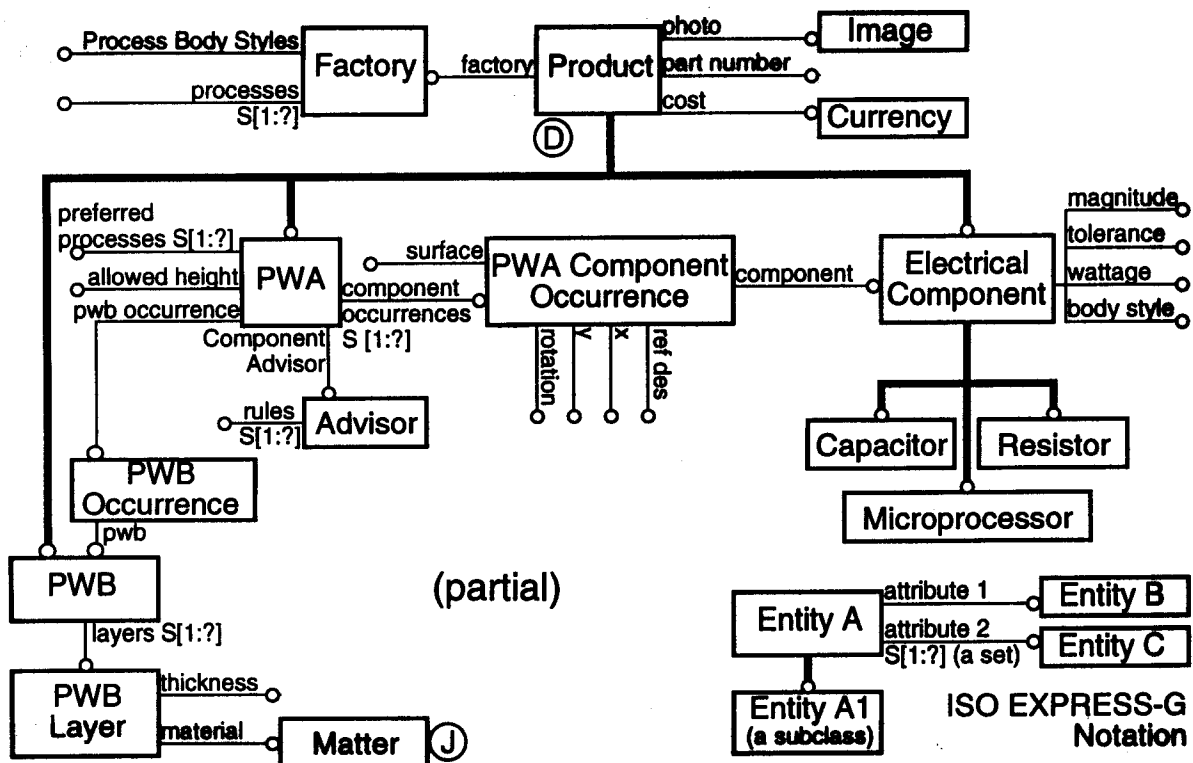


Figure E.1 Ad-hoc PWA Product Model

analysis model needs, ad-hoc extensions and combinations of these models have been made as needed.

The resulting simplified PWA product model used in this research is partially given in Figure E.1. The class Component Occurrence in this product model represents the physical usage of a component at a specific physical location in a PWA. It refers to a component-solder joint-PWB assembly. In contrast, the component attribute represents only a device (an Electrical Component) of a given part number which may be used many times on a given PWA. The unique identifier for a component occurrence is a reference designator (e.g. R110) versus a part number (e.g. PN 99120) for a component .

APPENDIX F

ANALYTICAL BUILDING BLOCKS

This appendix documents the preliminary general purpose analytical building blocks (GPABBs) introduced in Chapter 5. It also includes the object relationship diagram for the general ABB and PBAM representations defined in Chapters 5 and 6, as well for the case studies PBAMs in Chapter 9. While emphasis has been placed on analysis models needed for the PWA solder joint fatigue case study, attempts have been made to develop general representations that can be used by other applications as well. Note that the major focus of this research has been on the PBAM representation. Consequently the remaining analytical building blocks discussed here could be viewed as only a minor part of the total contribution. They were created to support the PBAM representation and are developed to just that extent.

F.1 Object Relationship Diagrams

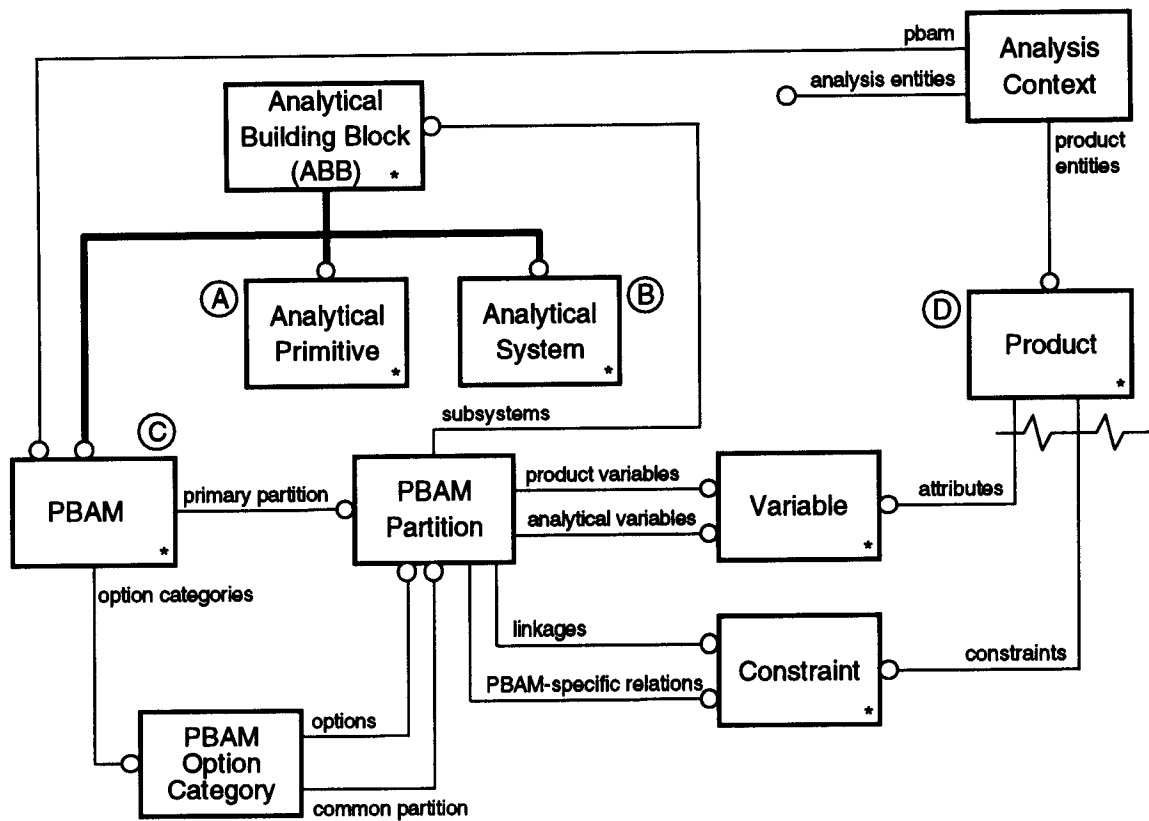
A surprisingly difficult, yet crucial, step in developing object-based representations is the creation of the object relationship diagrams (defined in Chapter 5) that show relationships between object types, including the is-a hierarchy on which inheritance of attributes and behavior is based. This Appendix gives a reasonable cut at a general hierarchy of mechanical engineering analysis models as object relationship diagrams by loosely using the EXPRESS-G notation (Appendix A). The hierarchy, only partially shown here, emphasizes classes needed by the case studies for solder joint fatigue analysis. The

reasoning behind each major section of the information model is given below along with relevant observations.

Encircled letters refer to the same entity in other figures, and asterisks (*) indicate entities implemented in the prototype to the degree required for the case studies. Capitalized terms in boxes denote class names, and plural attribute names indicate that the attribute is a collection consisting of the type of entity specified. (I.e., the official EXPRESS aggregation notation (e.g., SET[1:?]) has been left off to conserve space. For the purposes of this research, all such collections can be implemented as sets unless otherwise noted).

F.1.1 Associations Between Product Model and Analytical Model

Figure F.1 shows, at an abstract level, how product models and analysis models can be related. The philosophy taken here is that any physical object can only be represented by a *collection* of models (OBJECTIVES 4, 16, 23, 24)- there is no representation except the physical entity itself that contains full reality. Even the geometry of a product is represented by models of varying degrees of reality (e.g. electrical component body shapes in vendor catalogs, detailed component manufacturing drawings, and actual measured dimensions). Thus, the entity called Product is meant to represent reality but can do so only partially by serving as a composite object which holds relevant models of itself in a meaningful and usable manner. This enigma can be termed *aggregate reality* for future reference. It follows that analytical models are one part of the product model itself, so the phrase "product model-based analytical models" is somewhat misleading. However, it does convey the idea of basing analysis models on models of a "non-analysis" nature contained in the product model.



Note: This figure is in an earlier form that does not show ABB-related entities (e.g., ABB Partition) and how PBAM-related entities (e.g., PBAM Partition) are subclasses of them.

Figure F.1 Product Model-Analysis Model Associations

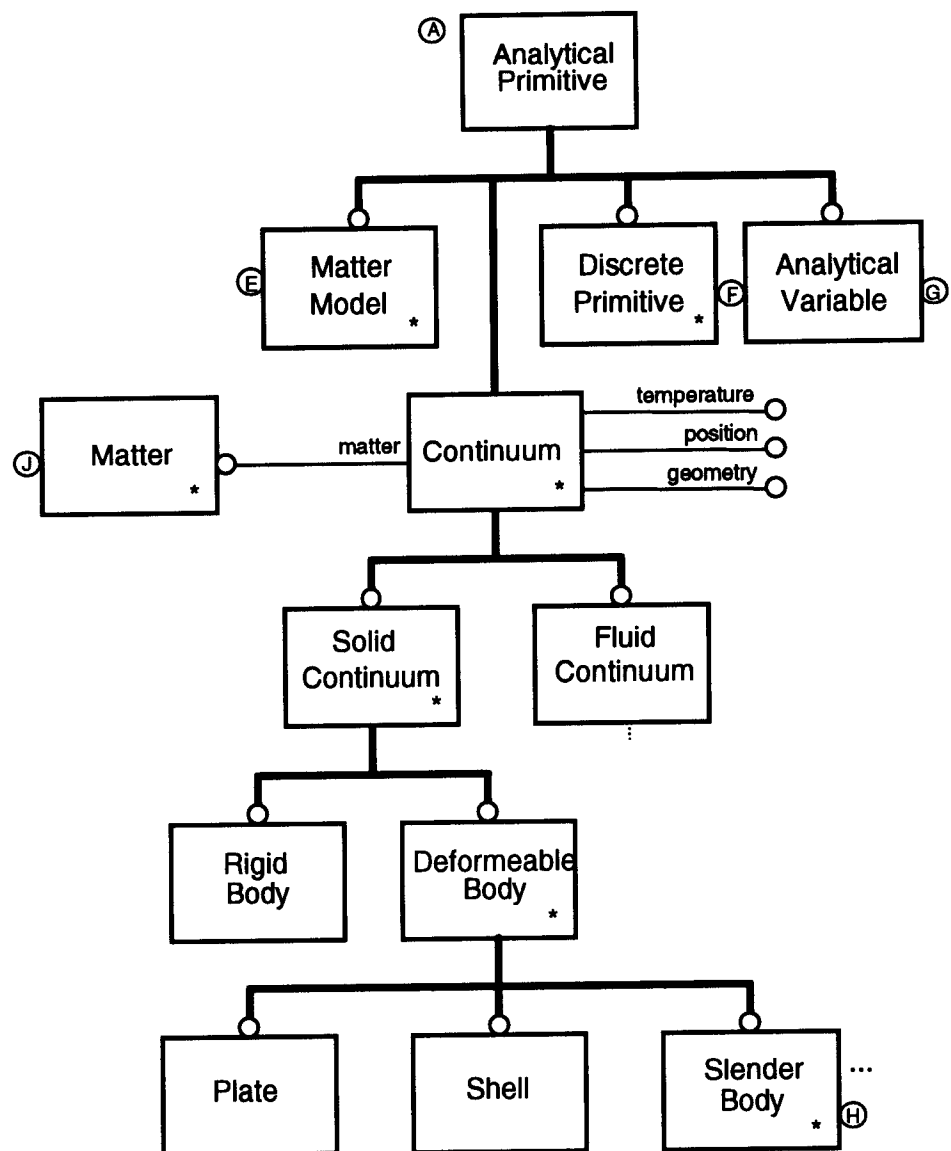


Figure F.2 Analytical Primitives

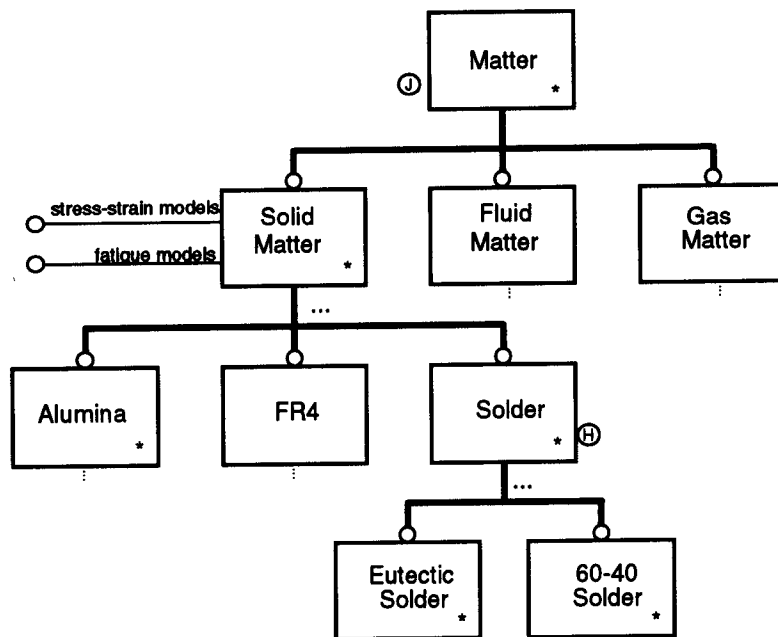


Figure F.3 Matter

F.1.2 Matter and Matter Models

Matter is defined as a physical substance without regard to a particular geometric bound and is categorized (very simply) based on room temperature behavior in Figure F.3. As **Matter** can be modeled in many different ways, each category of matter can be associated with many **Matter Models** (Figure F.4) analogous to the product/analysis aggregate reality discussion above. Therefore, **Matter** is *not* an Analytical Building Block; *it is a product-related entity* just as Component Occurrence is, and this figure is actually part of the product model object relationship diagram given in Appendix E.

Matter Models are GPABBs that capture properties of **Matter** with respect to given conditions (Figure F.4). Two classes of **Solid Matter Models** are given which are needed for solder joint fatigue analysis. Relating this approach to a similar one taken in STEP [ISO 10303-45] is one near term extension that could be investigated.

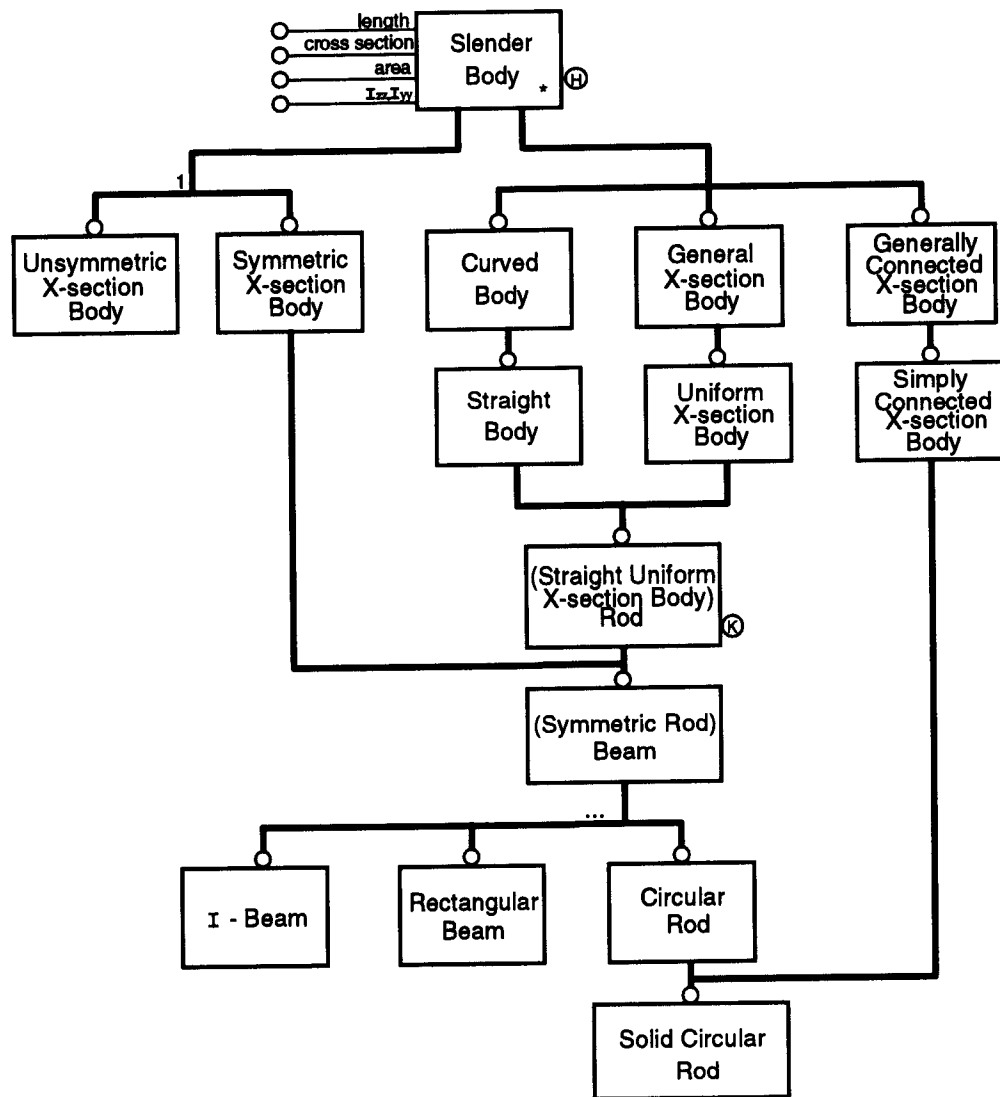


Figure F.5 Slender Bodies

The Deformable Body subclasses are decomposed according to geometrical characteristics because such factors drive major assumptions about the subclasses (e.g. a state of plane stress can be assumed if a body is thin).

The first level of Slender Body subclasses (Figure F.5) results from enumeration of several significant characteristics of slender bodies. The (Symmetric Straight Uniform Cross

Section Body) Beam class, which has a multiple inheritance relationship with several superclasses, has significant use in an Euler Beam System for bending. Note that an Analytical System is defined as the body *plus* loads, interconnections, etc.; therefore, an instance of any Continuum class generally will not be useful for analysis purposes apart from an Analytical System. A Circular Rod is a significant subclass for use in a simple torsional system. One could create a combinatorial explosion of Slender Body subclasses based on the first level of subclasses, but it is felt that only combinations of engineering significance which provide semantic footholds in the class hierarchy should be provided as pre-defined templates. Other subclasses could be derived as needed.

F.1.4 Discrete Primitives

The Discrete Primitive hierarchy (Figure F.6) is based heavily on bond graph concepts for discrete physical systems [Rosenberg and Karnopp, 1983; Ingrim, 1989a]. Translational mechanics primitives are shown at the bottom of each branch for reference; generally

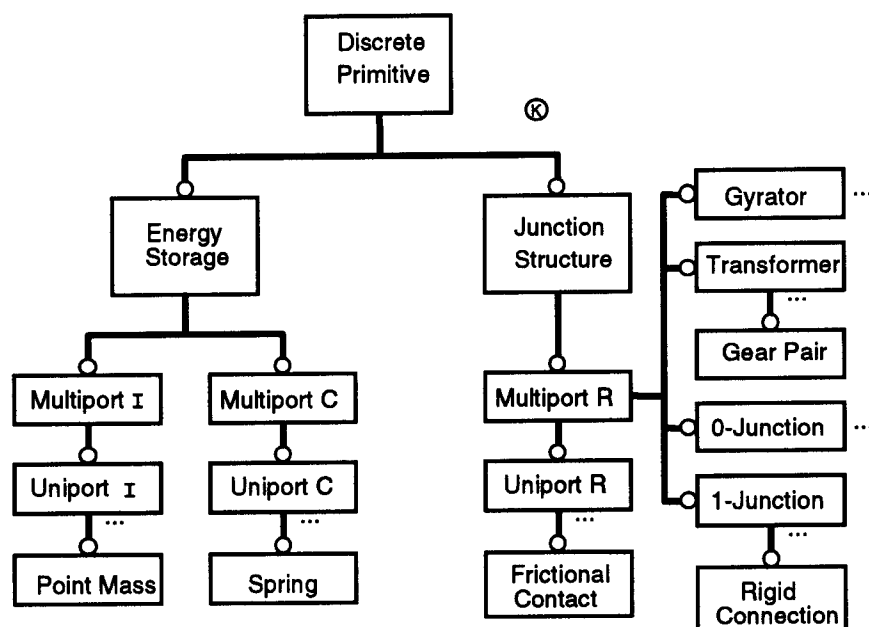


Figure F.6 Discrete Primitives

sibling classes also exist at each branch for other behavior domains as shown in Figure F.1.7 for Analytical Variables.

F.1.5 Analytical Variables

The categorization of Analytical Variables in Figure F.7 also is based on bond graph concepts. These variables are the loads and states that Analytical Systems can contain. STEP contains similar entities but does not group related variables as done here [ISO 10303-41]. Also operators associated with these variables could be captured in the information model.

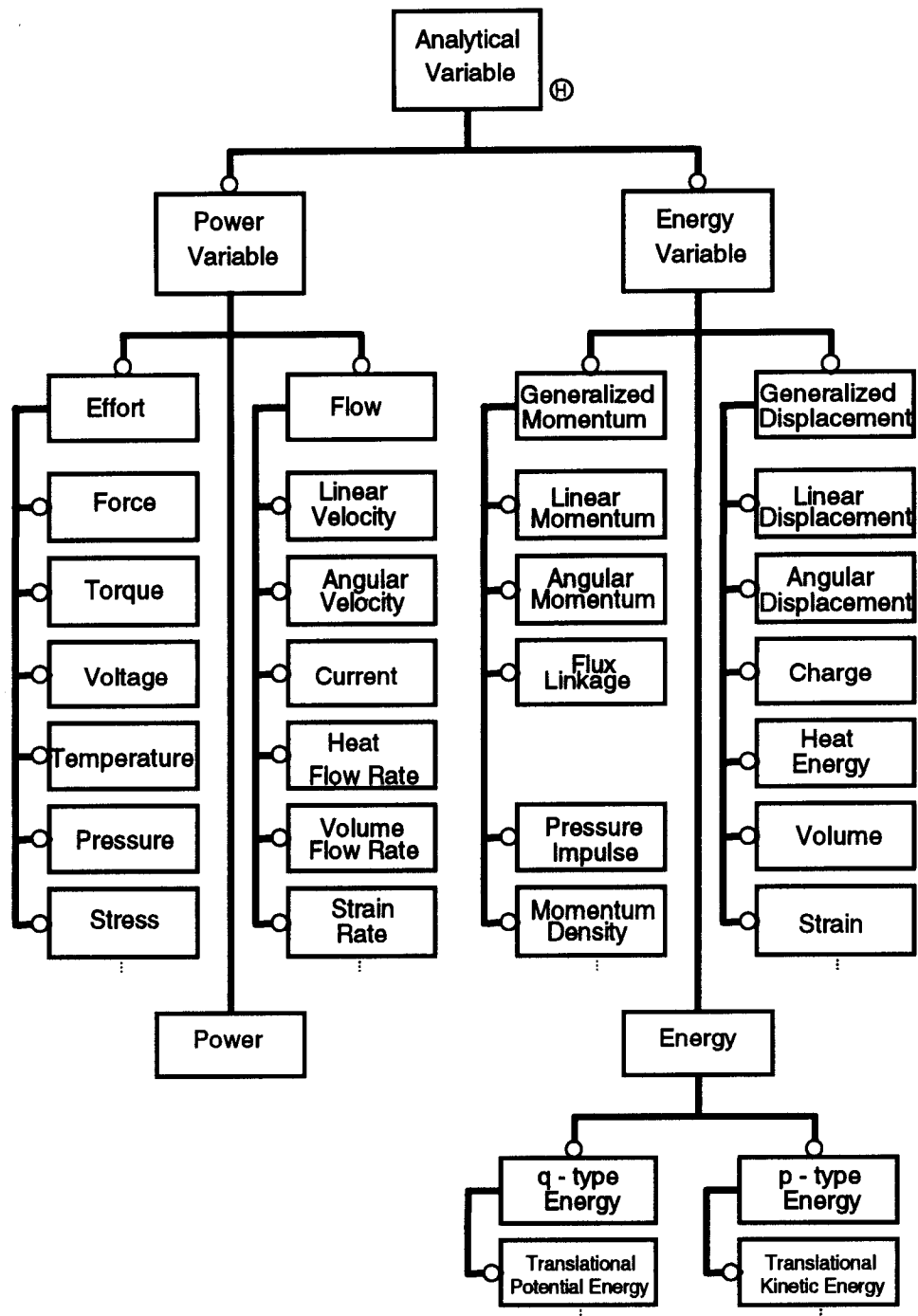


Figure F.7 Analytical Variables

F.1.6 Analytical Systems

Figure F.8 shows a few common Analytical Systems (Euler Beam Systems) from classical mechanics of materials. Again note the possibility of a large number of classes based on combinations of system characteristics. An instance of Junction Structure (from Figure F.6) indicates how analytical primitives and systems could be interconnected within an Analytical System.

The Interconnected Connected Rods System includes common operators that are needed by such a system (to date time varying operators have not been implemented). The attributes *rod1* and *rod2* are constrained to be Rods (Figure F.5); the other attributes could be similarly constrained.

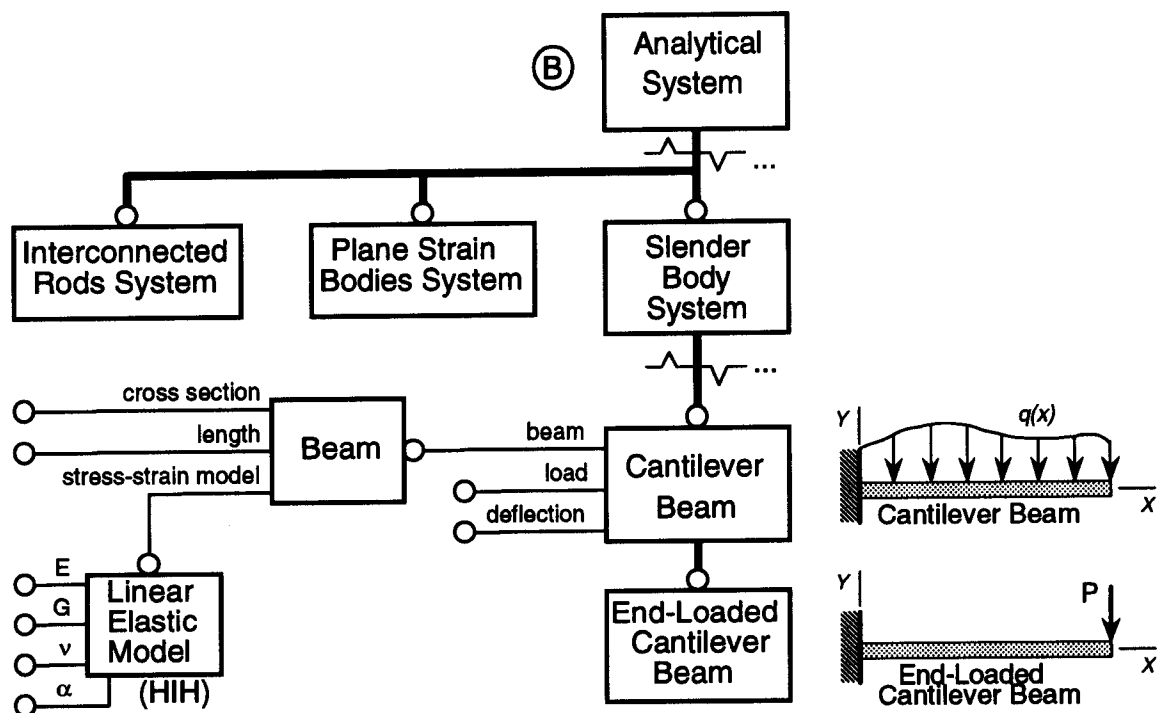


Figure F.8 Analytical Systems

F.1.7 Product Model-Based Analytical Models (PBAMs)

Figure F.9 illustrates the PBAMs needed for the solder joint case studies. Note that the PBAMs take the generic Analytical System and adapts them specifically for PWA applications. These PBAMs are further discussed in Appendix G and Chapter 9.

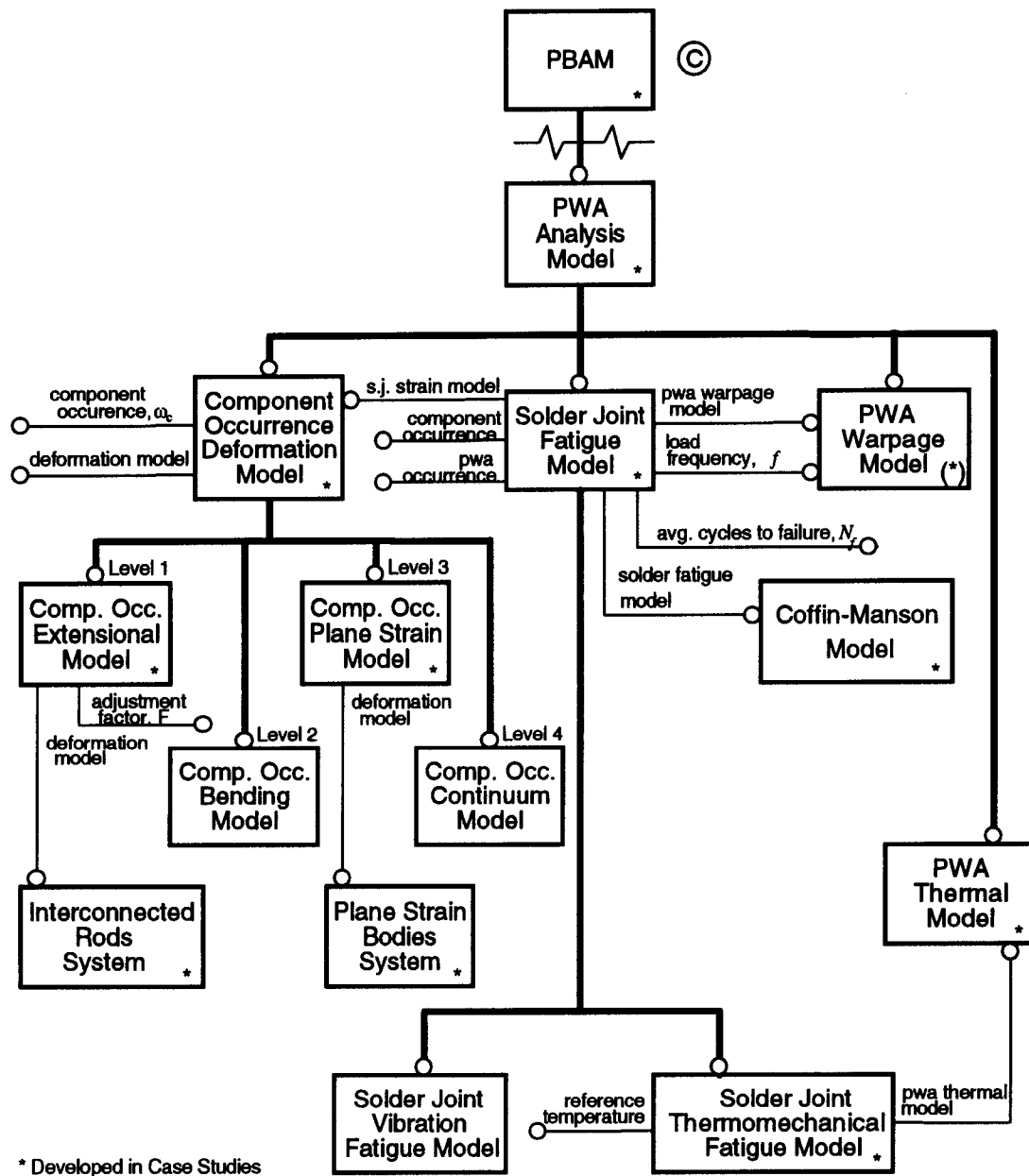


Figure F.9 PBAMs for PWA Analysis Applications

F.2 General Purpose ABB Datasheets

This portion of Appendix F contains ABB structural views (defined in Chapter 5) for some of the general purpose analytical building blocks (GPABBs) given in Section F.1 (however, not all views are given for each ABB). The aggregation of all ABB structural views for a particular type of ABB is known as the "datasheet" of that ABB, analogous to electrical component datasheets.

The object relationship diagrams in Section F.1 (in EXPRESS-G format) convey the is-a relationships of these entities more clearly, but additional aspects are included here (such as restrictions on attribute values); thus the different views of these GPABBs are complementary, just as with PBAMs. In some cases, the ABB structures in this section are specialized forms of the ABB structure. For example, the variables and relations in a **Fatigue Model** superclass have been divided into categories based on the semantics of that class of **Matter Models**. GPABB datasheets are like PBAM datasheets, except they have no product information linkages.

Fatigue Model (Abstract)

Superclass: Matter Model

Primary Partition

Variables

State Variables

average cycles to failure, \overline{N}_f

Material Properties

<see subclasses>

Material Property Coefficients

<see subclasses>

Conditions

mean cyclic temperature, \overline{T}

load frequency, f

ABB-Specific Relations

Constitutive Relations

<see subclasses>

Condition-Dependent Property Relations

<see subclasses>

Option Categories

<none extra>

Coffin-Manson Fatigue Model

Superclass: Fatigue Model

Primary Partition

Variables

State Variables

total cyclic strain range, $\Delta\epsilon$

elastic cyclic strain range, $\Delta\epsilon^e$

plastic cyclic strain range, $\Delta\epsilon^p$

Material Properties

fatigue strength exponent, b

fatigue strength coefficient, σ_f'

fatigue ductility exponent, c

fatigue ductility coefficient, ϵ_f'

Young's modulus, E

Conditions

<none extra>

ABB-Specific Relations

Constitutive Relations

1. $r(\Delta\epsilon, \Delta\epsilon^e, \Delta\epsilon^p)$

$$\Delta\epsilon = \Delta\epsilon^e + \Delta\epsilon^p$$

2. $r(\bar{N}_f, \Delta\epsilon^p, c, \epsilon_f')$

a. $\frac{\Delta\epsilon^p}{2} = \epsilon_f' (2\bar{N}_f)^c$

b. $\bar{N}_f = \frac{1}{2} \left(\frac{\Delta\epsilon^p}{2\epsilon_f'} \right)^{1/c}$

c. etc.

3. $r(\bar{N}_f, \Delta\epsilon^e, c, \sigma_f', E)$

a. $\frac{\Delta\epsilon^e}{2} = \frac{\sigma_f'}{E} (2\bar{N}_f)^b$

b. etc.

Condition-Dependent Property Relations

4. $r(E, f, \bar{T})$

5. $r(b, f, \bar{T})$

6. $r(\sigma_f', f, \bar{T})$

7. $r(c, f, \bar{T})$

8. $r(\epsilon_f', f, \bar{T})$

Note: These relations must be supplied by the matter being modeled.

Option Categories

<none extra>

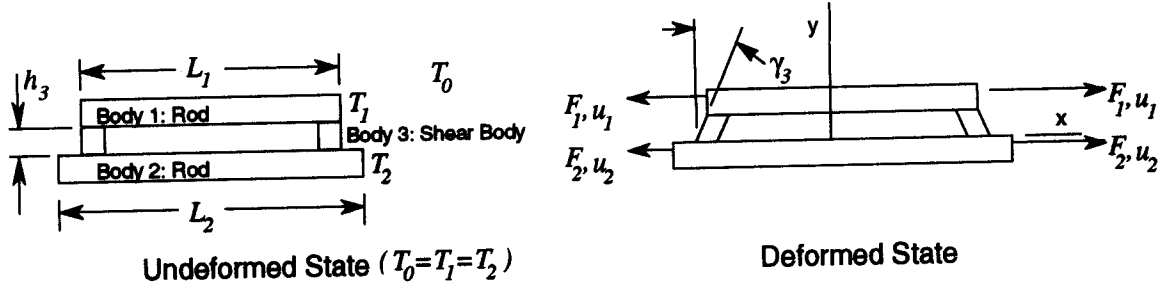
Subsystem Views

f, T -dependent Coffin-Manson Model	
<input type="radio"/> E	<input type="radio"/> c
<input type="radio"/> b	<input type="radio"/> ϵ_f'
<input type="radio"/> σ_f'	<input type="radio"/> $\Delta\epsilon$
<input type="radio"/> \bar{T}	<input type="radio"/> $\Delta\epsilon^e$
<input type="radio"/> f	<input type="radio"/> $\Delta\epsilon^p$
	<input type="radio"/> \bar{N}_f

(only one is given here, though others are possible)

Interconnected Rods System

Superclass: Analytical System



Primary Partition

Variables

reference temperature, T_o : Temperature
 steady state thermal expansion mismatch, $\Delta(\alpha\Delta T)$
 load yield factor, a

Subsystems

body 1: Rod

geometry: Regular Cylinder
 length, L_1
 stress-strain model: HIH Model
 Young's modulus, E_1
 coefficient of thermal expansion (CTE), α_1
 temperature, T_1
 strain, ϵ_1
 total deformation, u_1
 axial load, F_1

body 2: Rod

geometry: Regular Cylinder
 stress-strain model: HIH Model
 Young's modulus, E_2
 coefficient of thermal expansion (CTE), α_2
 temperature, T_2
 strain, ϵ_2
 total deformation, u_2
 axial load, F_2

body 3 (interface body): Shear Body

geometry: Rectangle
 height, h_3
 stress-strain model: HIH

shear modulus, G
yield stress, σ_Y
shear strain, γ_3

Semantic Linkages

<none extra>

ABB-Specific Relations

1. $r_1(\alpha_1, T_1, \alpha_2, T_2, \Delta(\alpha\Delta T), T_0)$
 $\Delta(\alpha\Delta T) = \alpha_2(T_2 - T_0) - \alpha_1(T_1 - T_0)$
2. $r_2(L_1, h_3, \gamma, \Delta(\alpha\Delta T))$

$$\gamma_3 = \frac{L_1 \Delta(\alpha\Delta T)}{2h_3}$$
3. $r_3(h_3, G, \sigma_Y, a, \Delta(\alpha\Delta T))$

$$a = \frac{\sigma_Y}{\sqrt{3}G} \frac{2h_3}{\Delta(\alpha\Delta T)}$$

Note: Relations 2 and 3 are for steady state conditions for the case where body 3 has a viscoplastic stress-strain model with F_1 and $F_2 = 0$. More general relations could be used if needed (e.g., to include in-plane deformation).

These relations really should not be represented explicitly as is done here, but should result from the fact that these bodies are connected together. Thus, in this *non-ideal* ABB, the bodies serve more as data storage objects as the relations inside them are not utilized.

Other Linkages

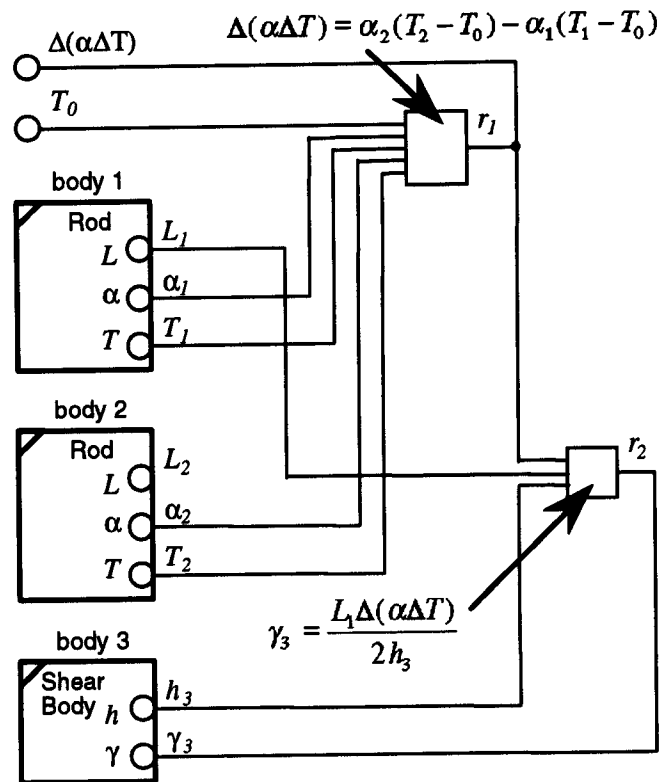
<none extra>

Option Categories

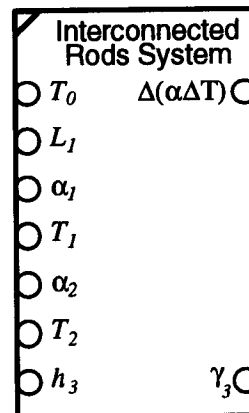
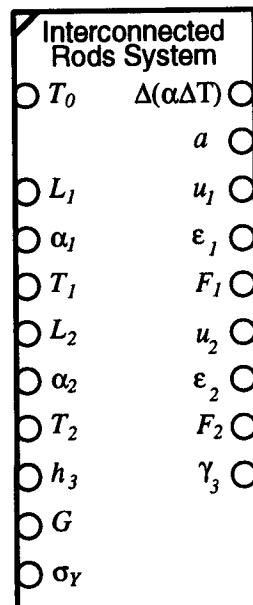
<none extra>

Constraint Schematic (partial)

Interconnected Rods System



Subsystem Views



Plane Strain Bodies System

Superclass: Analytical System

Primary Partition

Variables

reference temperature, T_o : Temperature
mesh density, d

Subsystems

body 1 : Plane Strain Body
 geometry : Rectangle
 length, L_1
 height, h_1
 stress-strain model : HIH Model
 elastic modulus, E_1
 Poisson's ratio, ν_1
 coefficient of thermal expansion (CTE), α_1
 temperature, T_1 : Temperature
body 2 : Plane Strain Body
 geometry: Rectangle
 length, L_2
 height, h_2
 stress-strain model : HIH Model
 elastic modulus, E_2
 Poisson's ratio, ν_2
 coefficient of thermal expansion (CTE), α_2
 temperature, T_2 : Temperature
body 3 (interface body) : Plane Strain Body
 temperature, T_3 : Temperature
 total shear strain extrema, $\gamma_{xy,extrema,3}$
 elastic shear strain extrema, $\gamma_{xy, extrema}^e, 3$
 plastic shear strain extrema, $\gamma_{xy, extrema}^p, 3$

Displacements and other state variables not shown are also possible for each body
Generally, one may want $p(\text{body}, x, y)$ and p extrema, where p is a state variable field (e.g., stress, strain, displacement).

Semantic Linkages

<none extra>

ABB-Specific Relations

1. Options [1.1] and [2.1]

Solution Method: FEA

(i.e., body3.geometry = Rectangle and body3.stress-strain model = HIIH Model)

$$\gamma_{xy,extrema,3} = f_{1.1}(\text{body1}(L_1, h_1, E_1, \nu_1, \alpha_1, T_1), \text{body2}(L_2, h_2, E_2, \nu_2, \alpha_2, T_2), \text{body3}(L_3, h_3, E_3, \nu_3, \alpha_3, T_3), T_0, d) \quad \text{etc.}$$

$$T_1 = f_{1.2}(\text{body1}(L_1, h_1, E_1, \nu_1, \alpha_1), \text{body2}(L_2, h_2, E_2, \nu_2, \alpha_2, T_2), \text{body3}(L_3, h_3, E_3, \nu_3, \alpha_3, T_3), T_0, \gamma_{xy,extrema,3}, d, e)$$

2. Options [1.2] and [2.1]

Solution Method: FEA

(i.e., body3.geometry = Parametric Shape A
and body3.stress-strain model = HIIH Model)

$$\gamma_{xy,extrema,3} = f_{2.1}(\text{body1}(L_1, h_1, E_1, \nu_1, \alpha_1, T_1), \text{body2}(L_2, h_2, E_2, \nu_2, \alpha_2, T_2), \text{body3}(L_3, h_3, L_{3a}, h_{3a}, r_3, s_3, E_3, \nu_3, \alpha_3, T_3), T_0, d) \quad \text{etc.}$$

$$T_1 = f_{2.2}(\text{body1}(L_1, h_1, E_1, \nu_1, \alpha_1), \text{body2}(L_2, h_2, E_2, \nu_2, \alpha_2, T_2), \text{body3}(L_3, h_3, L_{3a}, h_{3a}, r_3, s_3, E_3, \nu_3, \alpha_3, T_3), T_0, \gamma_{xy,extrema,3}, d, e)$$

3. Options [1.2] and [2.2]

Solution Method: FEA

(i.e., body3.geometry = Parametric Shape A
and body3.stress-strain model = BKH Model)

$$\gamma_{xy,extrema,3} = f_{3.1}(\text{body1}(L_1, h_1, E_1, \nu_1, \alpha_1, T_1), \text{body2}(L_2, h_2, E_2, \nu_2, \alpha_2, T_2), \text{body3}(L_3, h_3, L_{3a}, h_{3a}, r_3, s_3, E_3, \nu_3, \alpha_3, T_3), T_0, \lambda, \sigma_Y, d, a, n, e)$$

$$T_1 = f_{3.2}(\text{body1}(L_1, h_1, E_1, \nu_1, \alpha_1), \text{body2}(L_2, h_2, E_2, \nu_2, \alpha_2, T_2), \text{body3}(L_3, h_3, L_{3a}, h_{3a}, r_3, s_3, E_3, \nu_3, \alpha_3, T_3), T_0, \lambda, \sigma_Y, \gamma_{xy,extrema,3}, d, a, n, e) \quad \text{etc.}$$

Note: Only forms of the above relations where $\gamma_{xy,extrema,3}$ is the *output* have been implemented. Other forms will require an iterative solution.

Other Linkages

<none extra>

Option Categories

1. Body 3 Geometry

[1.1] Rectangle

[1.2] Parametric Shape A

Variables

<none extra>

Subsystems

body3

: Plane Strain Body

[1.1] geometry	: Rectangle
length, L_3	
height, h_3	
[1.2] geometry	: Parametric Shape A
length1, L_3	
length2, L_{3a}	
height1, h_3	
height2, h_{3a}	
radius, r_3	
shape, s_3	
volume, V_3	

2. Body 3 Stress-Strain Model

[2.1] HIH Model	(homogeneous, isotropic, hookean - linear elastic)
[2.2] BKH Model	(bilinear kinematic hardening - elastoplastic)

Variables

[2.2] load yield factor, a
[2.2] number of load steps, n
[2.2] convergence criteria, e

Subsystems

body 3	: Plane Strain Body
[2.1] stress-strain model	: HIH Model
elastic modulus, E_3	
Poisson's ratio, ν_3	
coefficient of thermal expansion (CTE), α_3	
[2.2] stress-strain model	: BKH Model
elastic modulus, E_3	
Poisson's ratio, ν_3	
coefficient of thermal expansion (CTE), α_3	
yield stress, σ_{Y3}	
strain hardening coefficient, λ_3	

Semantic Linkages

<none extra>

ABB-Specific Relations

<none extra>

Other Linkages

<none extra>

Subsystem Views

(see Component Occurrence Plane Strain Model constraint schematic)

APPENDIX G

CASE STUDY PBAMS

This Appendix contains the detailed PBAM views that were developed for the Case Studies. The PBAMs are presented in the following order, where indentation indicates the is-a relationship (see also in the PBAM object relationship diagram in Figure F.9)

PWA Analysis Model (Abstract)

Solder Joint Thermomechanical Fatigue Model (a.k.a. SJTF Model)

Component Occurrence Deformation Model (Abstract)

Component Occurrence Extensional Model (a.k.a. Extensional/Level 1 Model)

Component Occurrence Plane Strain Model (a.k.a. Plane Strain/Level 3 Model)

Basic PWA Thermal Model (in black box form only)

PWA Warpage Model (in constraint schematic form only)

Solder Joint Fatigue Model (Abstract)

Superclass: PWA Analysis Model

Primary Partition

Product Variables

- component occurrence, ω_c : PWA Component Occurrence
 - component
 - body style
 - solder joint
 - solder

Analytical Variables

- solder joint average cycles to failure, \overline{N}_f
- load frequency, f

Subsystems

- fatigue model
 - average cycles to failure, \overline{N}_f

Semantic Linkages

PBAM Variable

- solder joint average cycles to failure, \overline{N}_f

Subsystem Variable

- fatigue model
 - average cycles to failure, \overline{N}_f

PBAM-Specific Relations

<none extra>

Other Linkages

<none extra>

Option Categories

<none extra>

Solder Joint Thermomechanical Fatigue Model

Superclass: Solder Joint Fatigue Model

Primary Partition

Product Variables

component occurrence, ω_c
component
body style: leadless surface mount (LCCC, SMD chip, MELF, etc.)
solder joint
solder: 60/40 or Eutectic Solder

Analytical Variables

thermal load frequency, f
reference temperature, T_o
component temperature, T_c
substrate (PWB) temperature, T_s
mean cyclic solder joint temperature, \bar{T}_{sj}

Subsystems

fatigue model: Coffin-Manson Model
total cyclic strain range, $\Delta\epsilon$:= *invalid*
elastic cyclic strain range, $\Delta\epsilon^e$:= *invalid*
plastic cyclic strain range, $\Delta\epsilon^p$
load frequency, f
average cyclic temperature, \bar{T}
fatigue ductility exponent, c
fatigue ductility coefficient, ϵ_f

Semantic Linkages

PBAM Variable

thermal load frequency, f
mean cyclic solder joint temperature, \bar{T}_{sj}

Subsystem Variable

fatigue model
load frequency, f
average cyclic temperature, \bar{T}

PBAM-Specific Relations

$$1. \eta_1(\bar{T}_{sj}, T_o, T_c, T_s) \quad \bar{T}_{sj} = \frac{1}{4}(2T_o + T_c + T_s)$$

Other Linkages

fatigue model = component occurrence.solder joint.
solder.frequency and temperature dependent Coffin-Manson model

Option Categories

1. Strain Model Substitution

- [1.1] Component Occurrence Extensional Model
- [1.2] Beam Solder Joint Model
- [1.3] Plane Strain Solder Joint Model

Product Variables

component occurrence, ω_c

Analytical Variables

reference temperature, T_o

component temperature, T_c

substrate temperature, T_s

solder joint shear strain range, $\Delta\gamma_{sj}$

Subsystems

[1.1] Strain Model: Component Occurrence Extensional Model

[1.2] Strain Model: Beam Solder Joint Model

[1.3] Strain Model: Plane Strain Solder Joint Model

component model

temperature, T_c

substrate model

temperature, T_s

solder joint model

shear strain range, $\Delta\gamma_{sj}$

reference temperature, T_o

Fatigue Model

plastic cyclic strain range, $\Delta\epsilon^p$

Semantic Linkages

PBAM Variable

component deformation model

substrate deformation model

solder joint deformation model

reference temperature, T_o

solder joint shear strain range, $\Delta\gamma_{sj}$

Subsystem Variable

strain model.component model

strain model.pwb model

strain model.solder joint model

strain model.reference temperature, T_o

fatigue model.

plastic cyclic strain range, $\Delta\epsilon^p$

PBAM-Specific Relations

<none extra>

Other Linkages

component occurrence = strain model.component occurrence

2. Thermal Loading

[2.1] Thermal Cycling - uniform ΔT

[2.2] Power Cycling - temperatures based on power-up conditions

Product Variables

[2.2] pwa occurrence, ω_{pwa} : Product Occurrence

[2.2] component occurrence, ω_c

Analytical Variables

reference temperature, T_o

[2.1] steady state temperature, T_{ss}

[2.2] component temperature, T_c

[2.2] substrate temperature, T_s

Subsystems

[2.1] Thermal Model: none

[2.2] Thermal Model: PWA Thermal PBAM

pwa occurrence, ω_{pwa}

ambient temperature, T_a

component occurrence, ω_c

average component temperature, \bar{T}_c

average pwb temperature at component occurrence, $\bar{T}_{pwb @ \omega_c}$

Semantic Linkages

<none extra>

PBAM-Specific Relations

<none extra>

Other Linkages

[2.1] steady state temperature=component temperature

[2.1] steady state temperature=substrate temperature

[2.2] pwa occurrence=thermal model.pwa occurrence

[2.2] component occurrence=thermal model.component occurrence

[2.2] reference temperature=thermal model.ambient temperature

[2.2] component temperature=

thermal model.average component occurrence temperature

[2.2] substrate temperature=

thermal model.average pwb temperature at component occurrence

Option 3 Warpage Model Substitution

[3.1] Warpage Model : none

[3.2] Warpage Model : PWA Warpage Model

Solder Joint Fatigue
Design Parameter Model

Solder Joint Thermomechanical Fatigue Model	
<input type="radio"/> ω_c	<input type="radio"/> L_c
<input type="radio"/> T_o	<input type="radio"/> α_c
<input type="radio"/> T_{ss}	<input type="radio"/> α_s
<input type="radio"/> \bar{N}_f	<input type="radio"/> h_{sj}
<input type="radio"/> f	<input type="radio"/> F

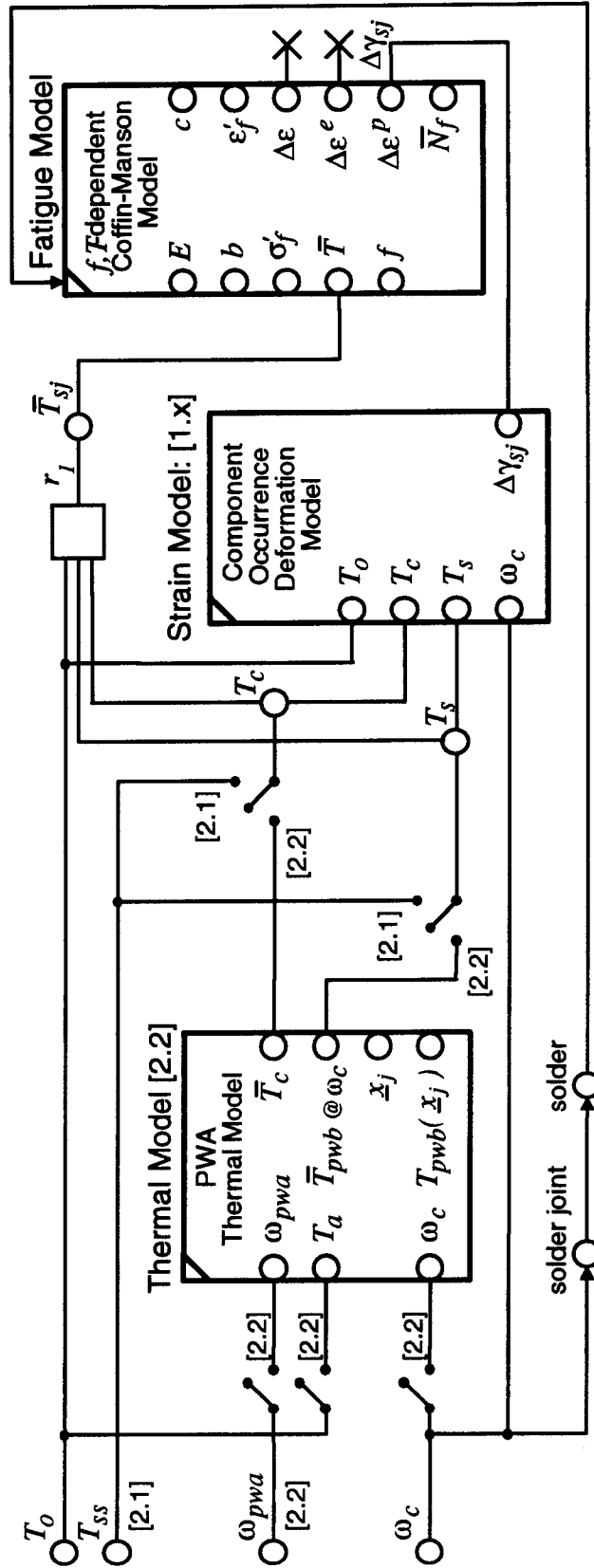
	T_o	T_{ss}	f	\bar{N}_f	ω_c	L_c	α_c	α_s	h_{sj}	F
1.	I	I	I	O	I	m	m	m	m	m
2.	I	I	I	I	I'	m	m	m	m	O
3.	I	I	I	I	I'	m	m	m	O	m
4.	I	I	I	I	I'	m	m	O	m	m
5.	I	I	I	I	I'	m	O	m	m	m
6.	I	I	I	I	I'	O	m	m	m	m

Solder Joint
Fatigue Model

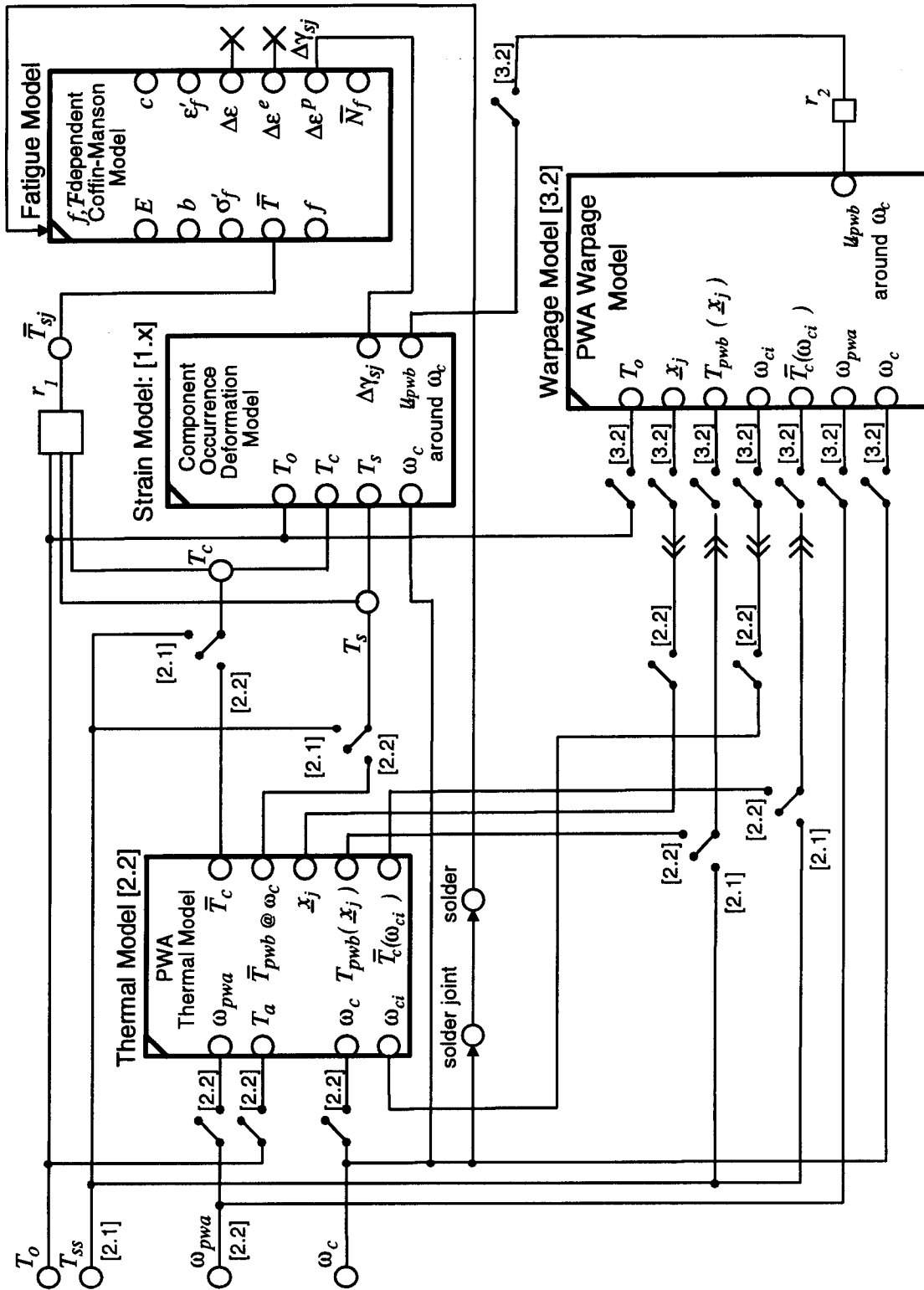
Solder Joint Thermomechanical Fatigue Model	
<input type="radio"/> ω_c	<input type="radio"/> \bar{N}_f
<input type="radio"/> T_o	<input type="radio"/> f
<input type="radio"/> T_{ss}	

	T_o	T_{ss}	f	\bar{N}_f	ω_c
1.	I	I	I	O	I
2.	I	I	O	I	I
*3.	I	O	I	I	I
*4.	O	I	I	I	I

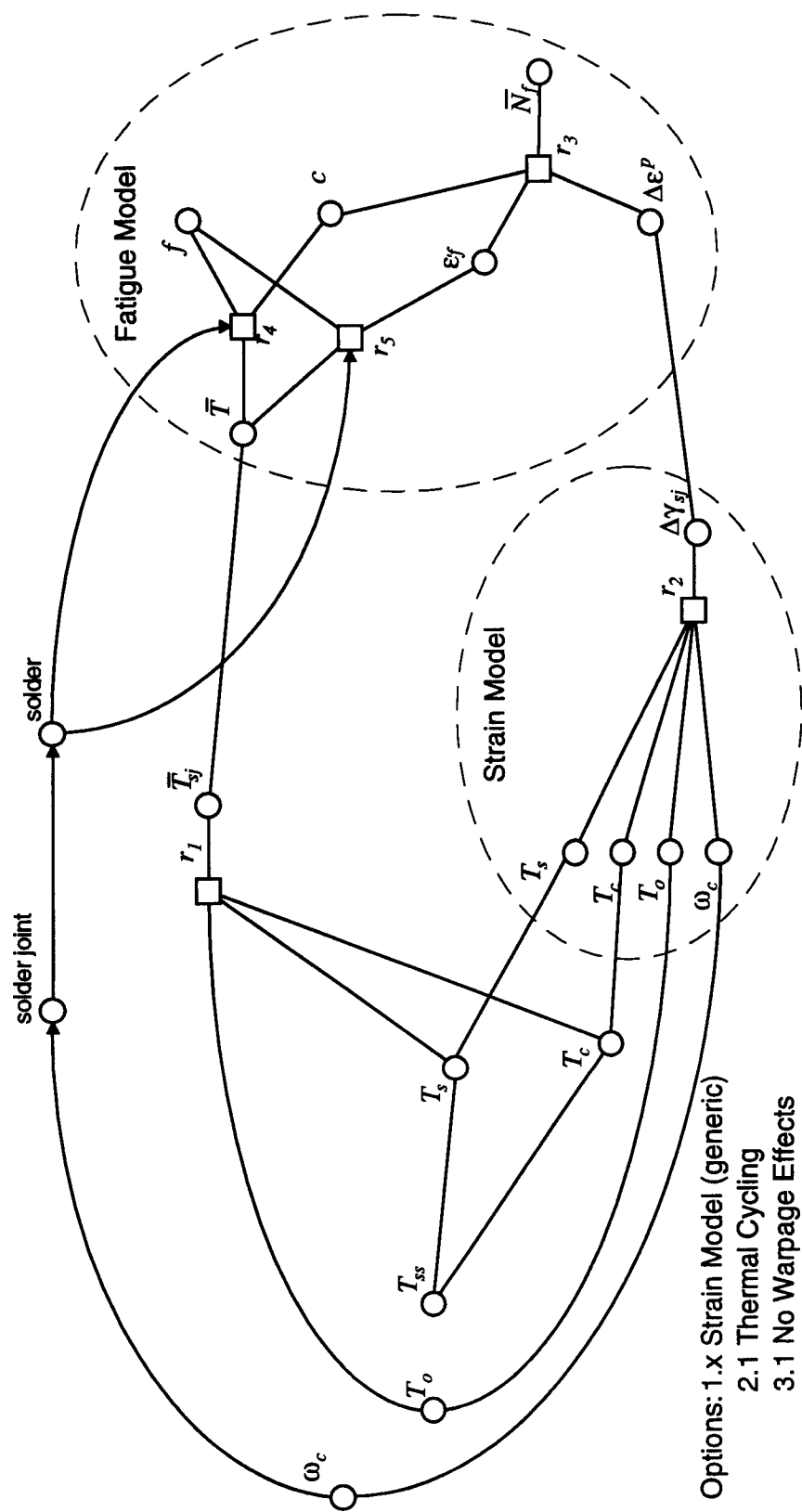
*Simultaneous equation solution



Constraint Schematic: Solder Joint Thermomechanical Fatigue Model (without Warpage Effects)



Constraint Schematic: Solder Joint Thermomechanical Fatigue Model (with Warpage Effects)



Extended Constraint Graph: Solder Joint Thermomechanical Fatigue Model under Thermal Cycling Load
(with generic Strain Model relation)

Component Occurrence Deformation Model (Abstract)

Superclass: PWA Analysis Model

Primary Partition

Product Variables

component occurrence, ω_c : PWA Component Occurrence
component : Electrical Component
primary CTE matter
HIH model*
coefficient of thermal expansion (CTE), α_c^*
body style : Leadless Surface Mount (LCCC, SMD chip, MELF, etc.)
total length, L_{total}^*
total width, w_{total}^*
pwb
primary CTE matter
HIH model*
coefficient of thermal expansion (CTE), α_s^*
solder joint

Other Analytical Variables

solder joint total cyclic shear strain range, $\Delta\gamma_{sj}$

Subsystems

Deformation Model: subclass-dependent

reference temperature, T_o
body 1
geometry
length, L_1
stress-strain model: HIH Model
coefficient of thermal expansion (cte), α_1
temperature, T_1
body 2
stress-strain model: HIH Model
coefficient of thermal expansion (cte), α_2
temperature, T_2

body3 (interface body)
geometry
height, h_3

* Analytical variables resulting from PATs

Semantic Linkages

PBAM Variable

component model

substrate model

solder joint model

Subsystem Variable

deformation model.body 1

deformation model.body 2

deformation model.body 3

PBAM-Specific Relations

1. approximate max. inter-solder joint distance, L_c (PAT)

$$L_c = L_{total} \quad \text{a. discrete surface mount components}$$

$$L_c = \sqrt{L_{total}^2 + w_{total}^2} \quad \text{b. rectangular surface mount chip carriers}$$

Other Linkages

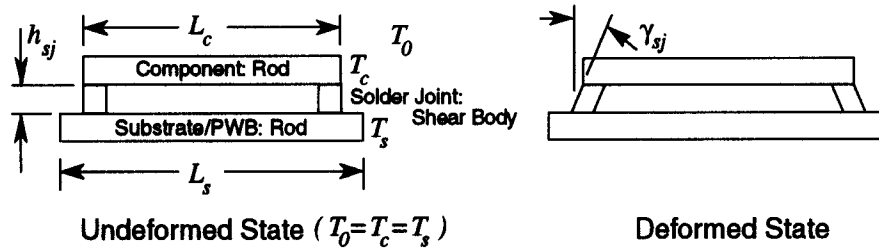
<none extra>

Option Categories

<none extra>

Component Occurrence Extensional Model

Superclass: Component Occurrence Deformation Model



Primary Partition

Product Variables

- component occurrence, ω_c
- solder joint
- solder
- HIH model*
- shear modulus, G_{sj}
- yield stress, σ_{sj}
- solder screen thickness, $t_{solder\ stencil}$

Other Analytical Variables

- adjustment factor, F

Subsystems

Deformation Model : Interconnected Rods System

- steady state thermal expansion mismatch, $\Delta(\alpha\Delta T)$

- load yield factor, a

- body1

- strain, ϵ_1

- total deformation, u_1

- body2

- strain, ϵ_2

- total deformation, u_2

- body3

- stress-strain model

- shear modulus, G_3

- yield stress, σ_{Y3}

- shear strain, γ_3

* Analytical variables resulting from PATs

PBAM-Specific Relations

$r_l(\text{solder joint type}, F)$

Solder Joint Type	F [Engelmaier, 1989]
SMD chip	0.7 - 1.2
castellated leadless	0.7 - 1.2
columnar leadless	1.0 - 1.5
leaded	1.0

Other Linkages

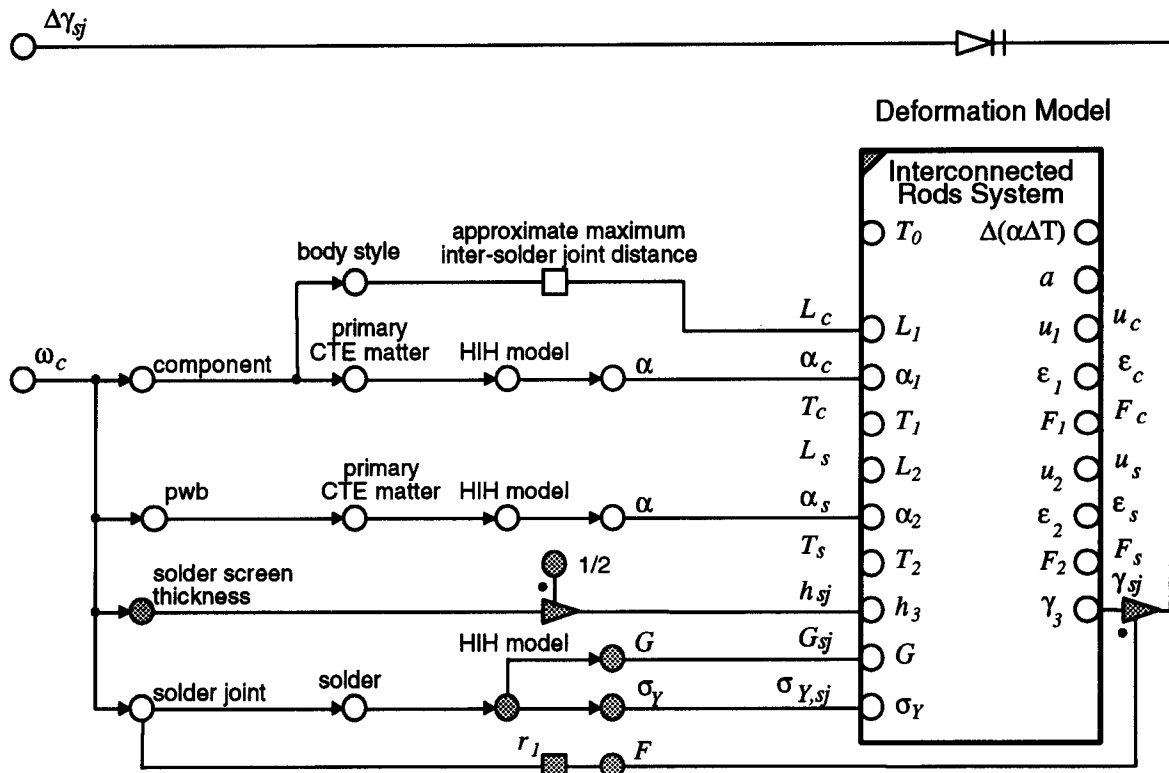
$$h_{sj} = \frac{1}{2} t_{\text{solder stencil}}$$

$$\Delta\gamma_{sj} = F |\gamma_{sj}|$$

Option Categories

<none extra>

Constraint Schematic

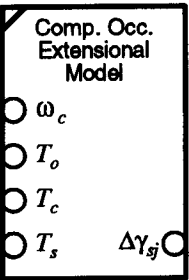


HIH = Homogeneous Isotropic Hookean (linear elastic)

Constraint Schematic: Component Occurrence Extensional Model

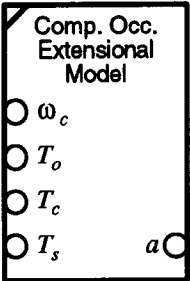
Subsystem Views & I/O Tables

Solder Joint Deformation Model



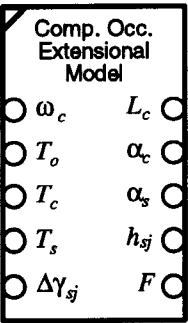
	T_o	T_c	T_s	$\Delta\gamma_{sj}$	ω_c
1.	I	I	I	O	I
2.	I	I	O	I	I
3.	I	O	I	I	I
4.	O	I	I	I	I

Load Step Estimator



	T_o	T_c	T_s	a	ω_c
1.	I	I	I	O	I
2.	I	I	O	I	I
3.	I	O	I	I	I
4.	O	I	I	I	I

Component Occurrence Design Parameter Model

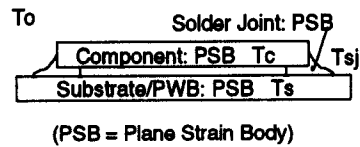


	T_o	T_c	T_s	$\Delta\gamma_{si}$	ω_c					
						L_c	α_c	α_s	h_{si}	F
1.	I	I	I	I	I'	m	m	m	m	O
2.	I	I	I	I	I'	m	m	m	O	m
3.	I	I	I	I	I'	m	m	O	m	m
4.	I	I	I	I	I'	m	O	m	m	m
5.	I	I	I	I	I'	O	m	m	m	m



Component Occurrence Plane Strain Model

Superclass: Component Occurrence Deformation Model



Primary Partition

Product Variables

component occurrence, ω_c

component

total height, h_c^*

primary CTE matter

HIH model*

Young's modulus, E_c

Poisson's ratio, ν_c

pwb

total thickness, t_s^*

primary CTE matter

HIH model*

Young's modulus, E_s

Poisson's ratio, ν_s

Other Analytical Variables

pwb model

length, L_s

Subsystems

Deformation Model : Plane Strain Bodies System

mesh density, d

body 1

geometry

height, h_1

stress-strain model: HIH Model

Young's modulus, E_1

Poisson's ratio, ν_1

body 2

geometry

* Analytical variables resulting from PATs

height, h_2
 stress-strain model: HIH Model
 Young's modulus, E_2
 Poisson's ratio, ν_2
 body 3 (interface body)
 temperature, T_3
 total shear strain extrema, $\gamma_{xy,extrema,3}$

Semantic Linkages

<none extra>

PBAM-Specific Relations

<none extra>

Other Linkages

$$L_s = 1.25 L_c$$

$$\Delta\gamma_{sj} = |\gamma_{xy,extrema,sj}|$$

$$T_{sj} = \frac{1}{2}(T_c + T_s)$$

Option Categories

1. Solder Joint Geometry

[1.1] Rectangle

[1.2] Detailed (SMD only)

Product Variables

component occurrence, ω_c

solder joint

[1.1] rectangular shape*

base length, L_b

standoff height, h_{sj}

[1.2] detailed shape*

base length, L_b

standoff height, h_{sj}

fillet height, h_f

fillet radius, r_{sj}

fillet shape s_f

volume, V_{sj}

Other Variables

<none extra>

Subsystems

Deformation Model : Plane Strain Bodies System

body 3 (interface body)

[1.1] geometry: Rectangle

length, L_3

height, h_3

[1.2] geometry: Parametric Shape A

length1, L_3

length2, L_{3a}

height1, h_3

height2, h_{3a}

radius, r_3

shape, s_3

volume, V_3

Semantic Linkages

<none extra>

PBAM-Specific Relations

<none extra>

Other Linkages

[1.1] $1/2(\omega_c.solder\ joint.rectangular\ shape.L_b) = deformation\ model.body\ 3.L_3$

[1.1] $\omega_c.solder\ joint.rectangular\ shape.h_{sj} = deformation\ model.body\ 3.h_3$

[1.2] $1/2(\omega_c.solder\ joint.rectangular\ shape.L_b) = deformation\ model.body\ 3.L_3$

[1.2] $1/2(\omega_c.solder\ joint.rectangular\ shape.L_b) = deformation\ model.body\ 3.L_{3a}$

[1.2] $\omega_c.solder\ joint.detailed\ shape.h_{sj} = deformation\ model.body\ 3.h_3$

[1.2] $\omega_c.solder\ joint.detailed\ shape.h_f = deformation\ model.body\ 3.h_{3a}$

[1.2] $\omega_c.solder\ joint.detailed\ shape.s_f = deformation\ model.body\ 3.s_3$

[1.2] $\omega_c.solder\ joint.detailed\ shape.V_{sj} = deformation\ model.body\ 3.V_3$

2. Solder Stress-Strain Model

[2.1] HIH Model

[2.2] BKH Model

Product Variables

component occurrence, ω_c

solder joint

solder

[2.1] HIH model

Young's modulus, E_{sj}

Poisson's ratio, ν_{sj}

[2.2] BKH model

Young's modulus, E_{sj}

Poisson's ratio, ν_{sj}

CTE, α_{sj}

yield stress, $\sigma_{Y,sj}$
strain hardening coefficient, λ_{sj}

Other Variables

[2.2] scale factor, k

Subsystems

Deformation Model : Plane Strain Bodies System

[2.2] number of load steps, n

[2.2] load yield factor, a

[2.2] convergence criteria, e

[2.1] stress-strain model: HHH Model

Young's modulus, E_3

Poisson's ratio, ν_3

[2.2] stress-strain model: BKH Model

Young's modulus, E_3

Poisson's ratio, ν_3

yield stress, $\sigma_{Y,3}$

strain hardening coefficient, λ_3

[2.2] Load Step Estimator : Component Occurrence Extensional Model

component occurrence, ω_c

reference temperature, T_o

component temperature, T_c

substrate temperature, T_s

deformation model

load yield factor, a

Semantic Linkages

<none extra>

PBAM-Specific Relations

<none extra>

Other Linkages

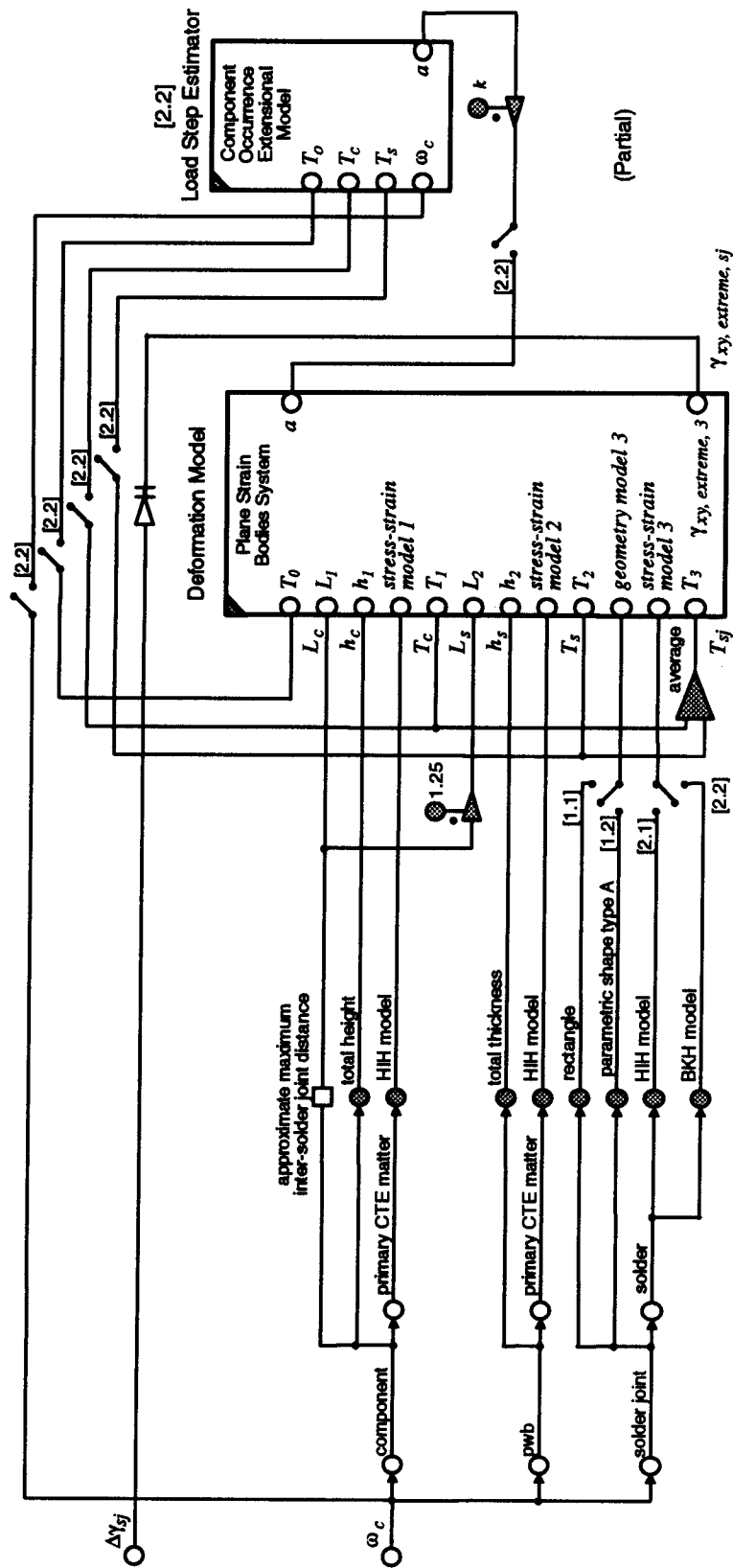
[2.2] load step estimator. $\omega_c = \omega_c$

[2.2] load step estimator. $T_o = T_o$

[2.2] load step estimator. $T_c = T_c$

[2.2] load step estimator. $T_s = T_s$

[2.2] load step estimator.deformation model. $a = k(\text{deformation model}.a)$



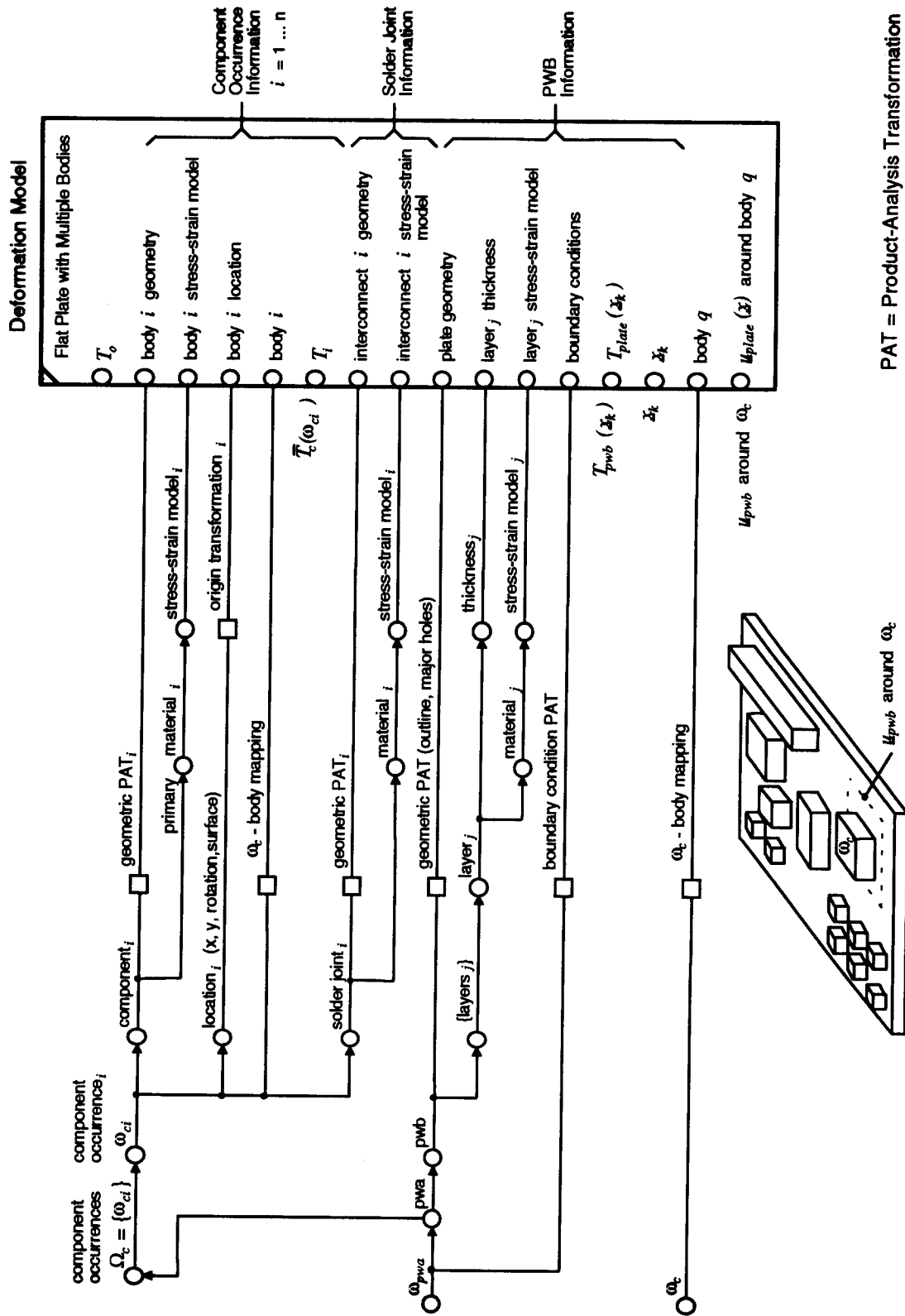
HIH = Homogeneous Isotropic Hookean (linear elastic), BKH = Bilinear Kinematic Hardening (elastoplastic)

Constraint Schematic: Component Occurrence Plane Strain Model (partial)

PWA Warpage Model

Superclass: PWA Analysis Model

(only constraint schematic given)

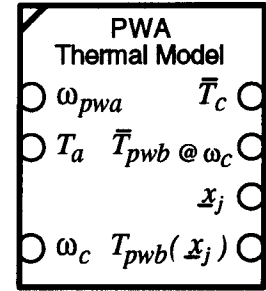


Constraint Schematic: PWA Warpage Model (conceptual PBAM)

PWA Thermal Model

Superclass: PWA Analysis Model

Note: This PBAM has been developed as a black box only. Only the bare essentials to provide input to the solder joint PBAMs are indicated (i.e., the internal relations are not developed).



Product Variables

component occurrence, ω_c
 pwa occurrence, ω_{pwa}
 parent assembly
 component occurrences, Ω_c
 pwb occurrence
 <incomplete>

Other Analytical Variables

ambient temperature, T_a
 component model
 average temperature, \bar{T}_c
 pwb model
 temperature, $T_{pwb}(\bar{x})$
 location, \bar{x}
 average temperature at component occurrence, $\bar{T}_{pwb} @ \omega_c$
 <incomplete>

Subsystems

<incomplete>

PBAM-Specific Relations

1. $r(T_o, \omega_{pwa}, \omega_c, T_c, T_s)$
 <incomplete>

Other Relations

<incomplete>

REFERENCES

- Anderson, E., 1989, *Knowledge FACTory II Manual*, Georgia Tech College of Computing, Atlanta, GA.
- An-Nashif, H., Powell, G. H., 1989, "A Strategy for Automated Modeling of Frame Structures" *Engineering with Computers*, Vol 5, pp 1-12.
- An-Nashif, H., Powell, G. H., 1991, "An Object-Oriented Algorithm for Automated Modeling of Frame Structures: Stiffness Modeling" *Engineering with Computers*, Vol 5, pp 1-12.
- Barr, A., and Feigenbaum, E. A., ed., 1981, *Handbook of Artificial Intelligence*, Vol I, William Kaufmann Inc., Los Altos, CA.
- Bartle, R. G., 1976, *The Elements of Real Analysis*, John Wiley & Sons, New York.
- Bender, E. A. and Williamson, S. G., 1991, *Foundations of Applied Combinatorics*, Addison-Wesley, New York.
- Bobrow, D. G., ed., 1984, *Qualitative Reasoning about Physical Systems*, MIT Press, Cambridge MA.
- Borning, A., Freeman-Benson, B., Maloney, J., Wilson, M., 1991 "Constraint Hierarchies and Their Applications," *1991 IEEE COMPCON Spring*, p. 388-393.
- Bras, B. A., 1992, *Foundations for Designing Decision-Based Design Processes*, Doctoral Thesis, U. Houston, Houston TX.
- Bravoco, R., Yadav, S., 1985, "Requirement Definition Architecture - An Overview," *Computers in Industry*, Vol 6, pp 237-251.
- Bravoco, R., Yadav, S., 1985, "A Methodology to Model the Functional Structure of an Organization," *Computers in Industry*, Vol 6, pp 345-361 (IDEF₀).
- Bravoco, R., Yadav, S., 1985, "A Methodology to Model the Information Structure of an Organization," *The Journal of Systems and Software*, Vol 5, pp 59-71 (IDEF_{1X}).
- Bray, O. H., 1988, *Computer Integrated Manufacturing - The Data Management Strategy*, Digital Press, Bedford MA.
- Breedveld, P. C., 1988, "Bond-Graph-based Model Generation," ASME WAM, *Automated Modeling for Design*, DSC Div. Vol 8, pp 41-47.
- Brown, D., Chandrasekaran, B., 1984, "Expert Systems for a Class of Mechanical Design Activity," *Knowledge Engineering in Computer-Aided Design*, Proc. IFIP WG5.2, Budapest Hungary, Sept. 17-19, 1984, pp 259-290.

- Cagan, J., Genberg, V., 1987, "PLASHTRAN: An Expert Consultant on Two Dimensional Finite Element Modeling Techniques," *Engineering with Computers*, Vol 2, pp 199-208.
- Catley, D., et al., 1991, "Computer-Aided Design Processes", Report of Committee V.5, 11th Intl. Ship and Offshore Structures Congress, China, pp. 181-247.
- CFI, Feb. 1991, "CFI '91 Design Representation Information Model and Programming Interface, Version 0.7," CAD Framework Initiative Inc., Austin TX.
- Chadha, B., Jazbutis, G., Wang, C. Y., Fulton, R. E., 1991, "An Appraisal of Modeling Tools and Methodologies for Integrated Manufacturing Information Systems," 1991 *ASME Computers in Engineering Conf.*, pp 19-27.
- Chandrasekaran, B., 1990, "Design Problem Solving: A Task Approach," *AI Magazine*, Winter, pp 59-71.
- Chandrasekaran, B., 1991, "Models versus Rules, Deep versus Compiled, Content versus Form - Some Distinctions in Knowledge Systems Research" *IEEE Expert*, April, pp 75-79.
- Charniak, E., McDermott, D., 1985, *Introduction to Artificial Intelligence*, Addison Wesley, Reading MA.
- Chen, P. P., ed., 1979, *Proc. Intl. Conf. on Entity-Relationship Approach to Systems Analysis and Design*, Los Angeles CA.
- Cormen T. H., Leiserson, C. E., Rivett, R. L., 1990, *Introduction to Algorithms*, MIT Press, Cambridge MA.
- Crandall, S. H., Dahl, N. C., and Lardner, T. J., 1978, *An Introduction to the Mechanics of Solid*, 2nd Ed. with SI Units, McGraw-Hill, New York.
- Davis, R., Shrobe, H., Szolovits, P., 1993 "What is a Knowledge Representation?," *AI Magazine*, Spring 1993, pp 17-33.
- Dieter, G. E., 1983, *Engineering Design - A Materials and Processing Approach*, McGraw-Hill, New York.
- Dixon, J. R., Duffey, M. R., Irani, R. K., Meunier, K. L., Orelup, M. F., 1988, "A Proposed Taxonomy of Mechanical Design Problems, " *Proc. ASME Computers in Engineering Conf.*, pp 41-46.
- Dodds, R. H., Rehak, D. R., Lopez, L. A., 1982, "Development Methodologies for Scientific Software," *Soft.-Pract. Exper.*, 12, Dec., pp. 1085-1100.
- Dym, C. L., Levitt, R. E., 1991, "Toward the Integration of Knowledge for Engineering Modeling and Computation," *Engineering with Computers*, Vol 7, pp 209-224.
- EIS, Oct. 1989, *Engineering Information Systems*, Vol I, II, III, Honeywell Systems & Research Center, U. S. A. F. Contract # F33615-87-C1401, Minneapolis MN.

Engelmaier, W., 1983, "Fatigue Life of Leadless Chip Carrier Solder Joints During Power Cycling," *IEEE Trans. on Components, Hybrids, and Manufacturing Technology*, Vol CHMT-6, No. 3, Sept. 1983, pp 232-237.

Engelmaier, W., 1989, "Thermal-Mechanical Effects," *Electronics Material Handbook. Vol 1 -Packaging*, Mingos, M. L., ed., ASM International, Materials Park, OH, pp 740-753.

Fennes, G. L., 1990, "Object-Oriented Programming for Engineering Software Development," *Engineering with Computers*, Vol 6, pp 1-15.

Filho, J. S. R. A, Devloo, P. R. B., 1991, "Object-Oriented Programming in Scientific Computations: The Beginning of a New Era," *Engineering Computations*, Vol 8, pp 81-87.

Fink, R. K., Callow, R. A., Larson, T. K., Ransom, V. H., 1988, "An Intelligent Modeling Environment for a Large Engineering Analysis Code: Design Issues and Development Experience," *Engineering with Computers*, Vol 3, pp 167-179.

Finn, D. P., Grimson, J. B., Harty, N. M., 1992, "An Intelligent Modelling Assistant for Preliminary Analysis in Design," *Artificial Intelligence in Design '92*, Gero, J. S., ed., pp. 579-596.

Forbus, in Bobrow, 1984.

Forde, B. W. R., Russell, A. D., Stierner, S. F., 1989, "Object-Oriented Knowledge Frameworks," *Engineering with Computers*, Vol 5, pp 79-89.

Forde, B. W. R., Stierner, S. F., 1989, "Knowledge-based Control for Finite Element Analysis," *Engineering with Computers*, Vol 5, pp 195-204.

Freeman-Benson, B. N., Borning, A., 1992, "Integrating Constraints with an Object-Oriented Language," *Proc. ECOOP '92*, Springer-Verlag, pp 268-286.

Freeman-Benson, B. N., Maloney, J., Borning, A., 1990 "An Incremental Constraint Solver," *Comm. ACM*, Vol 3, No 1, pp 54-63. Also avail. with Appendix as TR 89-08-06, Dept. of Comp. Sci. and Engineering, U. Washington, Seattle.

Freeman-Benson, B. N. and Wilson, M., 1990 "DeltaStar, How I Wonder What You Are: A General Algorithm for Incremental Satisfaction of Constraint Hierarchies" TR 90-05-02, Dept. of Comp. Sci. and Engineering, U. Washington, Seattle.

Friedman, L., 1991, *Comparative Programming Languages*.

Fromont, B., Sriram, D., 1992 "Constraint Satisfaction as a Planning Process," *Artificial Intelligence in Design '92*, pp 92-117.

Fruchter, R., Iwasaki, Y., Law, K. H., 1991, "Generating Qualitative Models for Structural Engineering Analysis and Design," *Proc. Computing in Civil Engineering & Symposium on Engineering Databases*, American Society of Civil Engineers, pp 268-267.

- Fulton, R. E., March 1987, "A Framework for Innovation," *Computers in Mechanical Engineering*, pp 26-40.
- Fulton, R. E., Craig, J. I., 1989, *Information Framework Technology for Integrated Design/Engineering Systems*, NSF Workshop Report, March 13-15, 1989, Callaway Gardens GA.
- Fulton, R. E., Ume, C., Scott, W. R., Jr., Hughes, J. L. A., Yeh, C. P., Peak, R. S., Sept. 1990, *Multidisciplinary Approach to Printed Wiring Board Design, First Report: Sept. 1988 - May 1990*, Manufacturing Research Center, Georgia Tech, Atlanta GA.
- Fulton, R. E., Ume, C., Gabertan, M. Y., Mao, J., Martin, T. L., Peak, R. S., M. S. Sedell, Yeh, C. P., Oct. 1991, *An Integrated Approach to Printed Wiring Board Design: Thermal Mechanical Behavior and Engineering Information Integration, Second Report: June 1990 - May 1991*, Manufacturing Research Center, Georgia Tech, Atlanta GA.
- Fulton, R. E., Ume, C., Gabertan, M. Y., Fu, C. Y., Mao, J., Martin, T. L., Peak, R. S., Tsang, F., Yeh, C. P., Oct. 1992, *An Integrated Approach to Printed Wiring Board Design: Thermal Mechanical Behavior and Engineering Information Integration, Final Report: June 1991 - September 1992*, Manufacturing Research Center - Advanced Electronic Packaging Lab, Georgia Tech, Atlanta GA.
- Fulton, R. E. and Peak, R. S., Oct. 1993, "Integrating Information Technology with Engineering Design Models," *First International Symposium on Research into Artifacts*, U. Tokyo, Japan, pp. 52-60.
- Garratt, J. D., Sept. 1993, *Prediction of Thermally Induced Printed Wiring Board Warpage*, Masters Thesis, Georgia Institute of Technology, Atlanta GA.
- GB, 1993, Call for Papers, *Inverse Problems in Engineering*, Gordon and Breach Science Pub., Cooper Station NY.
- Graver, J. O., 1992, *T-gen User's Guide*, Computer and Information Sciences Dept., University of Florida, Gainesville FL.
- Gray, F. L., 1989, "Thermal Expansion Properties," in *Electronics Material Handbook. Volume 1. Packaging*, Minges, M. L., ed., ASM International, Materials Park OH, pp 611-629.
- Goel, A. K., 1991, "Knowledge Compilation - A Symposium," *IEEE Expert*, April, pp 71-73.
- Guha, R. V., Lenat, D. B., 1990, "Cyc: A Midterm Report," *AI Magazine*, Fall, pp 32-59.
- Hinch, S. W., 1988, *Handbook of Surface Mount Technology*, John Wiley & Sons, New York.
- Hood, S. J., Palmer, E. R., Withers, D. H., 1989, "Automated Physical System Modelling Using Bond Graphs," *Computer Aided Design*, Vol 21, No. 9, pp 584-8.

Iannuzzelli, R., 1990, "Validation of Module Assembly Physical Models," *Proc. IEEE 40th CHMT Conf., Vol 2*, Las Vegas NV, pp 1071-1076.

IDEF0, June 1981, Report No. AFAWL-TR-81-4023, U.S. Air Force Wright Aeronautical Laboratories, Wright-Patterson Air Force Base, OH, .

IDEF1X, November 1985, "Integrated Information Support System (IISS)", Volume V - Common data model subsystem, Part 4 - Information Modeling Manual - IDEF1 Extended," Report No. AFAWL-TR-86-4006 U.S. Air Force Wright Aeronautical Laboratories, Wright-Patterson Air Force Base, OH.

IDEF1X, 1985, Information Modeling Manual IDEF1 - Extended (IDEF1X), Integrated Information Support System (IISS), ICAM Project Priority 6201, D. Appleton Co. Inc., Manhattan Beach CA.

Ignizio, J. P., 1991, *Introduction to Expert Systems*, McGraw-Hill, Inc.

Ingrim, M. E., Masada, G. Y., Beaman, J. J., 1988, "Bond Graph Representation of Thermomechanical Processes in One-Dimensional Thermoelastic Continua," *Automated Modeling for Design*, DSC Vol 8, ASME WAM.

Ingrim, M. E., Masada, G. Y., 1989a, "Extended Bond Graph Notation," Paper 89-WA/DSC-34, ASME WAM.

Ingrim, M. E., Masada, G. Y., 1989b, "Extended Bond Graph Representation of the Traction Problem in Linear Elastodynamics," Paper 89-WA/DSC-33, ASME WAM.

ISO 10303, 1992 "Guidelines for the Development and Approval of STEP Application Protocols, Version 1.0," ISO TC184/SC4/WG4 N34 (P5).

ISO 10303, 1991 "EXPRESS Usage Guide", ISO TC184/ SC 4/ WG5 N32.

ISO 10303-X, Industrial Auto. Sys. - Exchange of Prod. Model Data -

10303-1, 1992, Part 1: Overview.

10303-11, 1992, Part 11: The EXPRESS Language.

10303-12, 19xx, Part 12: The EXPRESS-I Language.

10303-21, 19xx, Part 21: Clear Text Encoding of the Exchange Structure

10303-22, 19xx, Part 22: STEP Data Access Interface

10303-31, 19xx, Part 31: Conformance Testing Methodology & Framework - General Concepts

10303-32, 19xx, Part 32: CTMF - Requirements on Testing Laboratories

10303-33, 19xx, Part 33: CTMF - Abstract Test Suite Specification

10303-34, 19xx, Part 34: CTMF - Abstract Test Methods

10303-41, 1991, Part 41: Fundamentals of Product Description and Support

10303-42, 1991, Part 42: Geometric and Topological Representation

10303-43, 1991, Part 43: Product Shape Interface Model

10303-44, 1991, Part 44: Product Structure Configuration Management

10303-45, 1990, Part 45: Materials

10303-46, 19xx, Part 46: Presentation
 10303-47, 19xx, Part 47: Shape Tolerances
 10303-48, 19xx, Part 48: Form Features
 10303-49, 19xx, Part 49: Product Life Cycle Support
 10303-101, 19xx, Part 101: Drafting
 10303-102, 19xx, Part 102: Ship Structures
 10303-103, 1989, Part 103: Electrical Applications.
 10303-104, 1991, Part 104: Finite Element Analysis
 10303-105, 1991, Part 105: Kinematics
 10303-201, 19xx, Part 201: Exchange of Explicit Drafting Models
 10303-202, 1990, Part 202: Exchange of Drawings with Reference to 3D Geometry
 10303-203, 1991, Part 203: Exchange of Configuration Controlled Data
 10303-204, 19xx, Part 204: Exchange of Boundary Representation Solid Models
 10303-205, 19xx, Part 205: Exchange of Sculptured Surfaces
 10303-206, 19xx, Part 206: Exchange of Wireframe Models
 10303-207, 19xx, Part 207: Sheet Metal Dies Planning and Design
 10303-208, 19xx, Part 208: Life Cycle Product Change Process
 10303-209, 19xx, Part 209: Composite and Metallic Structures

Jain, D., Krawinkler, H., Law, K. H., Luth, G. P., 1991, "A Formal Approach to Automating Conceptual Structural Design, Part I: Application to Floor Framing Generation," *Engineering with Computers*, Vol 7, pp 97-101.

Kim, C., O'Grady, P., Young, R. E., 1992 "A Large Scale, Multiple Constraint Network System for Design for Testability for Printed Wiring Boards," *Artificial Intelligence in Design '92*, Kluwer Academic Pub., Netherlands, pp 119-137.

Kiriyama, T., Oct. 1993, "A Design Environment for Micromachines" *First International Symposium on Research into Artifacts*, U. Tokyo, Japan, pp. 69-72.

Kiriyama, T., Tomiyama, and Yoshikawa, H., 1992, "Building a Physical Feature Database for Integrated Modeling and Design" *Sixth Intl. Workshop on Qualitative Reasoning About Physical Systems*, pp. 124-138.

Korngold, E. V., Shephard, M. S., Wentorf, R., Spooner, D. L., 1989, "Architecture of a Design System for Engineering Idealizations," *Proc. ASME Design Automation Conf.*, Vol. 1, pp. 259-265.

LaLonde, W., Pugh, J., 1990, *Inside Smalltalk - Volume I*, Prentice Hall.

Lassez, C., Aug. 1987, "Constraint Logic Programming," *Byte*, pp 171-176.

Lau, J. H., 1989, "Thermal Stress Analysis of SMT PQFP Packages and Interconnections," *Trans. ASME J. Electronic Packaging*, Vol 111, pp 2-8.

Lau, J. H., ed., 1991, *Solder Joint Reliability Theory and Applications*, Van Nostrand Reinhold, New York.

- Lau, J. H., ed., 1993 (in press), *Handbook of Thermal Stress and Strain in Microelectronics Packaging*, Van Nostrand Reinhold, New York.
- Lau, J. H., Rice, D. W., Avery, P. A., 1986, "Nonlinear Analysis of Surface Mount Solder Joint Fatigue," *Proc. IEEE CHMT Intl. Electronic Mfg. Technology Symposium*, San Francisco CA, Sept. 15-17, 1986, pp 173-184.
- Lee, H., Arora, J., 1991, "Object-Oriented Programming for Engineering Applications," *Engineering with Computers*, V7, pp 225-235.
- Leler, W., 1988, *Constraint Programming Languages - Their Specification and Generation*, Addison-Wesley, Reading MA.
- Liker, J., Fleischer, M., Arnsdorf, D., 1992, "Fulfilling the Promises of CAD," *Sloan Management Review*, Spring 1992, pp 74-86.
- Luth, G. P., Jain, D., Krawinkler, H., Law, K. H., 1991, "A Formal Approach to Automating Conceptual Structural Design, Part I: Methodology," *Engineering with Computers*, Vol 7, pp 79-89.
- Mackerle, J., Orsborn, K., 1988, "Expert Systems for Finite Element Analysis and Design Optimization - A Review," *Engineering Computations*, Vol 5, pp 90-102.
- Mackworth, A. K., 1977, "Consistency in Networks of Relations," *Artificial Intelligence*, Vol 8, pp 99-118.
- Maloney, J. H., 1991, *Using Constraints for User Interface Construction*, Doctoral Thesis, U. Washington, Seattle. Also avail. as TR 91-08-12, Dept. of Comp. Sci. and Engineering.
- Mäntylä, M., Sept. 1990, "A Modeling System for Top-Down Design of Assembled Products," *IBM J. Res. Develop.*, Vol 34, No. 5, pp 636-659.
- Mao, J. and Fulton, R. E., 1992, "Thermal Fatigue Reliability of the Solder Joints of Leadless Chip Components," in Fulton, Ume, et al., 1992.
- Mashburn, T. A., Anderson, D. C., 1991, "An Extensible Computer Environment for Modeling and Analysis in Mechanical Design," *Proc. ASME Computers in Engineering Conf.*, Vol 1., pp 127-135.
- MGC, 1991a, *Mentor Graphics System Overview Manual*, Mentor Graphics Corporation, Wilsonville, OR.
- MGC, 1991b, *Autotherm Reference Manual*, Mentor Graphics Corporation, Wilsonville OR.
- Miller, G. R., 1988, "A LISP-based Object-Oriented Approach to Structural Analysis," *Engineering with Computers*, Vol 4, pp 197-203.

- Mistree, F., Smith, W. F., Bras, B. A., Allen, J. K., and Muster, D., 1990, "Decision-Based Design: A Contemporary Paradigm for Ship Design," *Trans. Society of Naval Architects and Marine Engineers*, Jersey City NJ, 1990.
- Mittal, S., Dym, C. L., Mahesh, M., July 1986, "PRIDE: An Expert System for the Design of Paper Handling Systems," *Computer*, IEEE, pp 102-114.
- Mortenson, M. E., 1985, *Geometric Modeling*, Wiley & Sons, New York, pp 461-469.
- MSC, 1990, *MSC/XL User's Guide*, Version 2, MacNeal-Schwendler Corp., Los Angeles.
- NASB, 1977, *The New American Standard Bible*, Lockman Foundation, La Habra CA.
- Nijssen, G. M. and Halpin, T. A., 1989, *Conceptual Schema and Relational Database Design: A Fact Oriented Approach*, Prentice Hall, New York (NIAM).
- Nilson, D. R., 1991, "Integrated Software Tool Set for Design/Analysis of Electronic Hardware," *Proc. IEEE Annual Reliability and Maintainability Symposium*, pp 380-383.
- NIST, Sept. 1990, *The Initial Graphics Exchange Specification (IGES)*, Ver. 5.0, Reed, K., ed., NISTIR 4412, Gaithersburg MD.
- Niu, Q., Shephard, M. S., 1990, "Transfer of Solution Variables Between Finite Element Meshes," *SCOREC Report #4-1990*, Scientific Computation Research Center, Rensselaer Polytechnic Institute.
- NIV, 1984, *The Holy Bible, New International Version*, Intl. Bible Society, Zondervan Bible Publishers.
- ParcPlace, 1991, *Objectworks\Smalltalk User's Guide*, ParcPlace Corp., Palo Alto CA.
- Parisi, M. A., Rehak, D. R., 1986, "General Purpose Software for Probability Computations - A Virtual Machine Approach," *Engineering with Computers*, Vol 1, pp 161-174.
- Pahl, G., Beitz, W., 1988, *Engineering Design - A Systematic Approach*, The Design Council, London.
- PDES, 1991, *Abbreviated PDES Inc. Electrical/Electronic Project Technical Development Plan*, PMG018.01.00, PDES Inc., Charleston SC.
- Peak, R. S., Fulton, R. E., 1992a, "Selection of Printed Wiring Assembly Components Using a Multidisciplinary Integrated Information Framework," *Advances in Electronic Packaging 1992*, Chen, W. T. and Abe, H., ed., Proc. 1992 Joint ASME/JSME Conf. on Electronic Packaging, San Jose CA, pp 57-65.
- Peak, R. S. and Fulton, R. E., 1992b, "Integrating Analysis and Design Information in Electronic Packaging Using Product-Based Analytical Models," *Computer Aided Design in Electronic Packaging*, ASME WAM, EEP-Vol 3, Anaheim CA, pp 41-56.
- Peak, R. S. and Fulton, R. E., Nov. 1993, "Automating Routine Analysis in Electronic Packaging Using Product Model-Based Analytical Models (PBAMs); Part I: PBAM

Overview; Part II: Solder Joint Fatigue Case Studies," Papers 93-WA/EEP-23 and EEP-24, ASME Winter Annual Meeting, New Orleans LA, USA.

Peskin, R. L., Russo, M. F., 1988, "An Object-oriented System Environment for Partial Differential Equation Solution," *Proc. ASME Computers in Engineering Conf.*, pp 409-415.

Pinnel and Knausenberger, 1989, "Interconnection System Requirements," in *Electronics Material Handbook. Volume 1. Packaging*, Minges, M. L., ed., ASM International, Materials Park OH, pp 12-17.

Pound, R., 1989, "Conformal Coatings," *Electronics Material Handbook. Vol 1 - Packaging*, Minges, M. L., ed., ASM International, Materials Park, OH, pp 761-766.

Powell, G. H., An-Nashif, H., 1988, "Automated Modeling for Structural Analysis," *Engineering with Computers*, Vol 4, pp 173-183.

PTC, 1992, *Pro/ENGINEER Interface Guide*, Parametric Technologies Corp., Waltham MA.

PTC, 1993, *Pro/ENGINEER Model Users Guide*, Rel. 11.0, Parametric Technologies Corp., Waltham MA.

Puttre', M., Aug. 1992b, Virtual Prototypes Move Along Side Their Physical Counterparts, *Mechanical Engineering*, pp. 59-61.

Rangan, R. M., 1992, "An Object Oriented Dictionary Based CAD/CAM Data Exchange Environment," ASME WAM, Paper 92-WA/EDB-4.

Redfield, R., Mooring, B., 1988, "Concept Generation in Dynamic Systems using Bond Graphs," *Automated Modeling for Design*, DSC Vol 8, ASME WAM.

Rehg, J., Elfes, A., Talukdar, S., Woodbury, R., Eisenberger, M., Edahl, R. H., 1988, "Design Systems Integration in CASE," in *Expert Systems for Engineering Design*, Rychner, M. ed., Academic Press, pp 279-301.

Reusable Solutions, 1991, *FACETS User's Guide*, Reusable Solutions, Inc., Portland OR.

Rich, E., 1983, *Artificial Intelligence*, McGraw-Hill, New York.

Rinderle, J. R. and Colburn, E. R., 1990, "Design Relations," *Proc. ASME Design Theory and Methodology Conf.*, pp 267-272.

Roller, D., 1991, An Approach to Computer-Aided Parametric Design, *Computer-Aided Design*, Vol. 23, No. 5, pp. 385-391.

Rosen, D. W., 1992, *A Feature-Based Representation to Support the Design Process and the Manufacturability Evaluation of Mechanical Components*, Doctoral Thesis, University of Massachusetts.

Rosenberg, R. C., Karnopp, D. C., 1983, *Introduction to Physical System Dynamics*, McGraw Hill, New York.

- Rosenberg, R. C., Redfield, R., ed., 1988, *Automated Modeling for Design*, DSC Vol 8, ASME WAM.
- Rumbaugh, J., Blaha, M., Premerlani, W., Eddy, F., Lorensen, W., 1991, *Object-Oriented Modeling and Design*, Prentice Hall, Englewood Cliffs NJ.
- Ryan, S. A., 1992, "PDES/STEP: A Status Report," *Engineering Data Management: Key to Integrated Product Development*, Chase, T. R., ed., *Proc. ASME Intl. Computers in Engineering Conf.*, San Francisco CA, pp 43-47.
- Ryder, F. L., 1961, *Creative Engineering Analysis*, Prentice-Hall, Englewood Cliffs, NJ.
- Salustri, F. A., Venter, R. D., 1992 "An Axiomatic Theory of Engineering Design Information," *Engineering with Computers* 8, pp 197-211.
- Sapossnek, M., 1989, "Research on Constraint-Based Design Systems," *Proc. 4th Intl. Conf. on Applications of A. I. in Engineering*, pp 385-403.
- Sargent, R. G., 1985, "An Expository On Verification and Validation of Simulation Models," *Winter Simulation Conference*.
- SASI, 1990, *ANSYS User's Guide*, Swanson Analysis Systems Inc., Houston PA.
- SASI, 1993, "ANSYS/ProFEA. Feature-Based Parametric Optimization," *ANSYS News*, 3rd Iss., Swanson Analysis Systems Inc., Houston PA, pp. 8-14.
- Schorr, H., Rappaport, A., eds., 1989, *Innovative Applications to Artificial Intelligence*, AAAI Press/MIT Press.
- SDRC, 1990, *I-DEAS Electronics Design Automation (EDA) Interface Utility Guide*, Structural Dynamics Research Corp., Milford, OH.
- SDRC, 1991, *I-DEAS Bi-Directional EDA Interface User's Guide*, Structural Dynamics Research Corp., Milford, OH.
- Serrano, D. and Gossard, D., 1988, "Constraint Management in MCAE," *A. I. in Engineering Design*, Vol 1, Gero, J. S., ed., Elsevier, New York, pp 217-240.
- Shephard, M. S., 1988, "The Specification of Physical Attribute Information for Engineering Analysis," *Engineering with Computers*, Vol 4, pp 145-155.
- Shephard, M. S., Baehmann, P. L., LeCoz, Y. L., Sham, T. L., 1992 "Methodology for the Integration of Global/Local Thermal and Thermo-Mechanical Analyses of Multichip Modules," *Computer Aided Design in Electronic Packaging*, ASME WAM, Anaheim CA, EEP-Vol 3, pp 65-72.
- Shephard, M. S., Finnagan, P. M., 1988, "Integration of Geometric Modeling and Advanced Finite Element Preprocessing," *Finite Elements in Analysis and Design*, Vol 4, No. 2, pp 147-161.

- Shephard, M. S., Korngold, E. V., and Wentorf, R., 1990, "Design Systems Supporting Engineering Idealizations," *Geometric Modeling for Product Engineering*, Wozny, M. J. et al., ed., Elsevier Science B. V. North-Holland, pp. 279-299.
- Silverman, B. G., Murray, A. J., 1991, "Full-Sized Knowledge-Based Systems Research Workshop," *AI Magazine*, Vol 11, No. 5, pp 88-94.
- SMT CON, April 1991, 2nd Annual SMT CON Conf. and Exposition, *Surface Mount Technology* magazine, sponsor, Intl. Electronics Packaging Society, endorser, Atlantic City NJ.
- SMT DFM, Oct. 1992, "Surface Mount Technology Design for Manufacturability", Society of Mfg. Engineers, course sponsor, Minnetonka, Minnesota.
- Steinberg, D. S., 1988, *Vibration Analysis for Electronic Equipment*, John Wiley & Sons, New York.
- Stephens, E. R., 1993, *LEGEND: Laboratory for the Generation, Evaluation, and Navigation of Design*, Doctoral Thesis, Georgia Institute of Technology, Atlanta GA.
- STI, 1993, *STEP Programmers Tool Kit*, STEP Tools Inc., Troy NY.
- Sutherland, I., 1963, "SKETCHPAD: A Man-Machine Graphical Communication System," *Proc. Spring Joint Computer Conf.*, IFIPS, pp. 329-345.
- Ver Planck, D. W. and Teare, Jr., B. R., 1954, *Engineering Analysis*, J. Wiley & Sons, New York.
- Walton, M., 1986, *The Deming Management Method*, Perigee Books, Putnam Pub. Co., New York.
- Wentorf, R., Budhiraja, A., Collar, R. R., Shephard, M. S., Baehmann, P. L., 1992 "Two Prototypes Testing the Use of an Expert System in the Control of Structural Analysis Idealizations," *Expert Systems for Scientific Computing*, Elsevier Science Publisher B. V., pp 283-326.
- Whelan, P. T., 1989, *CAD/CAE Database Management System Requirements for Mechanical Parts*, Doctoral Thesis, Georgia Institute of Technology, Atlanta GA.
- Wilson, P. R., ed., 1993, "EXPRESS Tools and Services," Rensselaer Design Research Center Report 93022.
- Winston, P. H., 1984, *Artificial Intelligence*, 2nd ed., Addison-Wesley, Reading MA.
- Wirfs-Brock, R., Wilkerson, B., Wiener L, 1990, *Designing Object-Oriented Software*, Prentice Hall.
- Wolfram, S., 1991, *Mathematica - A System for Doing Mathematics by Computer*, 2nd ed., Addison-Wesley, New York.

- Yagawa, G., Mochizuki, Y., Yoshimura, S., 1991, "Automated Structural Design Based on Expert's Knowledge and Fuzzy Control," *Proc. 1991 ASME Computers in Engineering*, Vol 1, pp 23-28.
- Yang, Y., 1991, "Three-Schema Implementation in STEP: Standard for the Exchange of Product Model Data," ASME Computers in Engineering Conf., *Engineering Databases: An Engineering Resource*, pp 1-5.
- Yeh, C. P., 1992, *An Integrated Information Framework for Multidisciplinary PWB Design*, Doctoral Thesis, Georgia Institute of Technology, Atlanta GA.
- Yeh, C. P., Banerjee, K., Martin, T. L., Ume, C., Fulton, R. E., May 1991a, "Experimental and Analytical Investigation of Thermally Induced Warpage for PWBs," *Proc. of IEEE 41st Electronic Component Technology Conference (ECTC)*, Atlanta GA.
- Yeh, C. P., Fulton, R. E., Peak, R. S., 1991c, "A Prototype Information Integration Framework for Electronic Packaging," Paper 91-WA-EEP-43, *ASME WAM*, Atlanta GA.
- Yeh, C. P., Fulton, R. E., Ume, C., Hughes, J. L. A., 1990, "A Multidisciplinary Approach to PWB Design," Paper 90-WA/EEP-43, *ASME WAM*, Dallas TX.
- Yeh, C. P., Ume, C., Fulton, R. E., Dec. 1991b "Correlation of Analytical and Experimental Approaches for PWB Thermal/Mechanical Design," *ASME WAM*, Atlanta GA.
- Yeh, C. P., Ume, C., Fulton, R. E., 1993 (in press), "Experimental and Analytical Investigation of Thermally Induced Warpage for Circuit Boards," in *Handbook of Thermal Stress and Strain in Microelectronics Packaging*, J. H. Lau, ed., Van Nostrand Reinhold, New York.
- Yoshimura, S., Yagawa, G., Mochizuki, Y., 1990, "An Artificial Intelligence Approach to Efficient Fusion First Wall Design," *Lecture Notes in Computer Science (Cooperative Product Development)*, Springer-Verlag, pp. 502-521.

VITA

*I believe that God made me for a purpose ... He also made me fast.
And when I run, I feel His pleasure.*

Erick Liddell
Chariots of Fire

*Whatever you do, do your work heartily, as for the Lord rather than for men;
Knowing that from the Lord you will receive the reward of the inheritance.
It is the Lord Christ whom you serve.*

The Apostle Paul
Colossians 3:24-25 [NASB]

Russell Speights Peak was born in Georgia in the United States of America. He was reborn¹ in 1974, and began truly living² in 1980. He married Haruko Kinoshita, B. Che. '84, whom he knew from their undergraduate years at Georgia Tech. They have been blessed with two children, Joy and Jonathan.

The author is the product of a diverse secondary education, which included two high schools in Texas and one in Georgia. His university education is less diverse, having received all his degrees in Mechanical Engineering, and all from the Georgia Institute of Technology: a Bachelors degree in 1984, a Masters degree in 1985, and, with this thesis, a Doctorate degree in 1993.

He worked for AT&T Bell Laboratories in Middletown, New Jersey as a Physical Designer of business telephones from the mid to late 1980s. His AT&T career also included a brief manufacturing internship at AT&T Shreveport Works in Louisiana, after which he returned to Georgia Tech to begin his doctoral program. Having completed his doctorate, he plans to join the Mechanical Engineering Research Laboratory of Hitachi, Ltd. as a Visiting Researcher in Tsuchiura, Ibaraki, Japan.

¹ John 3:3, I Corinthians 6:9, 10, 11

² John 17:3, II Corinthians 5:17, Galatians 5:16-26, 22-23