WARNING CONCERNING COPYRIGHT RESTRICTIONS

The Copyright Law of 1976 (United States Code, Title 17) governs the making of photocopies or other reproductions of copyrighted material.

Under certain conditions specified in the law, libraries and archives are authorized to furnish a photocopy or other reproduction. One of these specific conditions is that the photocopy or reproduction is not to be "used for any purpose other than private study, scholarship, or research." If a user makes a request for, or later uses, a photocopy or reproduction [including that made by electronic transmission of reserve material] for purposes in excess of "fair use," that user may be liable for copyright infringement.

Copyright source:
Annex B (normative)

EXPRESS-G: A Graphical Subset of EXPRESS

B.1 Introduction

EXPRESS-G is a graphical notation for the display of information models. Although it has been specifically developed for the graphical rendition of information models defined in the EXPRESS language, it may be used as a modeling technology in its own right. The notation only supports a subset of the EXPRESS language.

The design goals for the notation are:

- The diagrams should be intuitively understandable.
- The diagrams should support levels of model abstraction.
- A diagram must be able to span more than one sheet of paper.
- The pictures should be definable using minimal computer graphics capabilities. Further, it should be possible to print the diagrams using only non-graphic symbols, for example on a line printer.
- It should be possible to develop a processor that automatically converts from an EXPRESS textual description to the graphical description3).

An information model is considered to consist of definitions of things (entity, type, function etc.), constraints on things, relationships between definitions, and constraints on the relationships. Using the EXPRESS language, an information model is represented by “sentences” in the language. Using EXPRESS-G, an information model is represented by graphic symbols forming a diagram. There are three types of symbols:

Definition: These symbols denote the things (i.e concepts, ideas, etc) which form the basis of the information model.

Relationship: These symbols describe the relationships which exist among the definitions.

Composition: These symbols enable a model diagram to be displayed on more than one sheet of paper.

EXPRESS-G supports the notions of entity, type, relationship and cardinality. It also separately supports the notion of schema. It does not provide any support for the constraints and constraint mechanisms provided by the EXPRESS language.

B.1.1 Graphic line styles

Three styles of lines are used by EXPRESS-G — a thin solid line, a thick solid line, and a dashed line — which shall be readily distinguishable. For displays that support graphics there should be no problems in choosing suitable line styles. For displays that only support a single line width, thick lines can be drawn as two closely spaced parallel lines. For line printer type displays, the lines have to be drawn using characters rather than graphics. Example character style graphics are given in Clause B.9.

B.1.2 Model forms

An EXPRESS-G model can take one of several forms, depending on whether a single schema or multiple schemas are being represented and on the amount of detail being displayed. These will be discussed in more detail later but for now it is noted that an EXPRESS-G model is either

a) A single schema with only the definitions and relations within the schema being displayed (an entity-level model), or

b) Multiple schemas, where only the schemas and schema-schema relations are displayed (a schema-level model).

B.1.3 Example model

Before going into the details of the EXPRESS-G notation, Figure B 1 and Figure B 2 show an entity level EXPRESS-G model for the single EXPRESS schema given in Example 120 (see Clause B.10).

![Diagram of EXPRESS-G model for Example 120]

Figure B 1 – Complete entity-level model of Example 120 (Page 1 of 2).

For illustrative purposes, the graphical model has been spread over two “pages” to show cross-page referencing. The EXPRESS-G model contains most of the notational symbols described later and is intended to provide a sample context for the more detailed exposition.

![Diagram of EXPRESS-G model for Example 120](Page 2 of 2)

The model is mainly about the concept of person. A person has certain defining characteristics,
such as first and last names (and maybe a nickname), date of birth, and some description of their hair. A person is either male or female. A male may have a female wife; a female may have a male husband and may have a maiden name. A person may have children, who, of course, are also persons. There are some constraints on husband and maiden name. There are also some constraints on male and female. The definitions of things within the model are denoted by the boxes, and relationships among the definitions are denoted by the lines joining the boxes. The idea lying behind the differing line styles displayed is that the weight of the line gives an indication of the “strength” of the relationship.
B.2 Definition symbols

All the definition symbols consist of a rectangular box enclosing the name of the thing being defined. The type of the definition is distinguished by the line style of the box. The notation provides symbols for the EXPRESS simple types, defined types, entity, and schema.

B.2.1 EXPRESS simple type symbols

The EXPRESS language includes some predefined simple types, namely BINARY, BOOLEAN, INTEGER, LOGICAL, NUMBER, REAL, and STRING. These are the terminals of the language. The symbol for these is a rectangular solid box with a double vertical line at the right hand end of the box, and the name of the type is enclosed within the box, as shown in Figure B 3.

![Simple type symbols](image)

Figure B 3 – Simple type symbols.

One additional type, GENERIC, is defined in the EXPRESS language, and this is also treated as a terminal of EXPRESS-G.

B.2.2 Type symbols

The symbols for the three forms of TYPE definitions, namely SELECT, ENUMERATION and defined data type, are all dashed boxes and are shown in Figure B 4.

![Type definition symbols](image)

Figure B 4 – Type definition symbols.

The symbol for a defined data type consists of a dashed box enclosing the name of the TYPE.

The symbol for a SELECT type, as shown in Figure B 4, consists of a dashed box with a double vertical line at the left hand end, enclosing the name of the SELECT.

The symbol for an ENUMERATION type, as shown in Figure B 4, consists of a dashed box with a double vertical line at the right hand end, enclosing the name of the ENUMERATION. Although an ENUMERATION is not a terminal of the EXPRESS language (because its definition includes the enumerated things), it is a terminal of the EXPRESS-G language.
B.2.3 Entity symbol

Figure B 5 illustrates the symbol for an ENTITY, which has a rectangular solid form enclosing the name of the ENTITY.

![entity](image)

Figure B 5 – Entity definition symbol.

B.2.4 Function and procedure symbols

EXPRESS-G does not support any notation for either FUNCTION or PROCEDURE definitions.

B.2.5 Rule symbol

EXPRESS-G does not support any notation for a RULE definition. However, as discussed in Clause B.5.3, the names of entities that are parameters in a RULE may be flagged with an asterisk.

B.2.6 Schema symbol

The symbol for a SCHEMA is shown in Figure B 6. It is a rectangular solid box with the name of the SCHEMA in the upper half, which is divided from the lower half of the box by a horizontal line. The lower half of the symbol shall be empty.

![schema](image)

Figure B 6 – Schema definition symbol.
B.3 Relationship symbols

Related definition symbols are connected via lines of different types as shown in Figure B 7.

```
<table>
<thead>
<tr>
<th>Normal line</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dashed line</td>
</tr>
<tr>
<td>Thick line</td>
</tr>
</tbody>
</table>
```

Figure B 7 - Relationship Line Styles

A relationship for an Optional attribute of an Entity shall be displayed as a dashed line. An schema-schema Reference shall be displayed as a dashed line. A Tree relationship (i.e., a Supertype and Subtype relation) shall be displayed as a thick solid line. All other relationships shall be displayed as normal width solid lines.

Relationships are bidirectional, but, following the EXPRESS style, one of the two possible directions is emphasized. For example, if an Entity A has an explicit attribute which is Entity B, then the emphasized direction is from A to B. In EXPRESS-G, the “to” end of a relationship shall be marked with an open circle.

![Diagram](image)

Figure B 8 - Partial entity-level model illustrating relationship directions from Example 121.

(Page 1 of 1)

NOTE 16 - Relationship directions are illustrated in Figure B 8, which is an incomplete rendition of the EXPRESS code given in Example 121. The model consists of two entities, from_ent and to_ent. From_ent has to_ent as an optional attribute and the simple REAL type as a required attribute. In turn, entity to_ent has a defined data type called strings as a required attribute, and strings is composed of the terminal STRING.
B.4 Composition Symbols

Graphical model representations are large and can span several pages. Each page in a model shall be numbered. The symbols for achieving inter-page references are shown in Figure B 9. Likewise, a schema may utilise definitions from another schema. The symbols for inter-schema references are also shown in Figure B 9.

Page references

<table>
<thead>
<tr>
<th>page#, ref# (#, #, ...)</th>
<th>reference onto this page</th>
</tr>
</thead>
<tbody>
<tr>
<td>page#, ref# name</td>
<td>reference onto another page</td>
</tr>
</tbody>
</table>

Inter-schema references

<table>
<thead>
<tr>
<th>schema.def</th>
<th>definition Referenced from another schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>alias</td>
<td>definition Used from another schema</td>
</tr>
</tbody>
</table>

Figure B 9 – Composition symbols

B.4.1 Page references

Where there is a relationship between definitions on separate pages, the relationship line on the two pages shall be terminated by an oval (rounded) box that contains a page number and a reference number, as shown in Figure B 9. The page number is the number of the page where the to definition resides. The reference number is used to distinguish between multiple references onto a page. The composition box on the from page additionally shall contain the name of the to definition. The reference oval on the to page may additionally contain a parenthesised list of the page numbers of the from pages which refer to this reference.

NOTE 17 – The use of page referencing can be seen by looking at the person entity and the date definitions in Figure B 1 and Figure B 2. The oval labelled 2,1 from the person indicates that the definition is to be found on page 2 of the model as reference 1. On page 2 of the model, the oval symbol reference into date indicates that this definition is referenced from some other definition on another page of the model. The number in parentheses indicates that the referencing item is to be found on page 1 of the model.
B.4.2 Inter-schema references

Some definitions that are utilised in a schema may actually be defined in another schema. Inter-schema references shall be indicated by a rounded box enclosing the name of the definition qualified by the schema name, as also shown in Figure B.9.

Definitions that are accessed from another schema via an EXPRESS USE statement may be enclosed by a solid rectangular box. If the definition is aliased, then the alias name may be placed within the box below the oval.

Definitions that are accessed from another schema via an EXPRESS REFERENCE statement may be enclosed by a dashed rectangular box. If the definition is aliased, then the alias name may be placed within the box below the oval.

NOTE 18 — An illustration of the use of inter-schema references may be found later in Figure B.12.
B.5 Entity level model

An Entity-level model is an EXPRESS-G model that represents the definitions, and their relationships, that comprise a single schema. Thus, the components of such a model consist of simple, type, entity and relationship symbols, together with role and cardinality information as appropriate.

B.5.1 Role names

In EXPRESS the attributes of an ENTITY are role named. The text string representing the role name may be placed on the relationship line connecting an ENTITY symbol to its attribute definition symbols.

B.5.2 Cardinalities

The attributes of Entities and user defined types can be aggregates (i.e., LIST, SET, BAG, and ARRAY). In EXPRESS, an aggregation is denoted by a phrase like LIST [1:?] OF ... In EXPRESS-G, any aggregation specification is indicated on the relationship line, following the role name (if any). To conserve space, only the first letter of the aggregate type shall be used and the OF shall be omitted. Note that if there is no aggregation specified then the cardinality is exactly one for a required relation and zero or one for an optional relation.

NOTE 19 — For example, the EXPRESS model given in Example 121 is fully displayed in EXPRESS-G in Figure B 10.

![Figure B 10 – Complete entity-level model of Example 121. (Page 1 of 1)](image)

B.5.3 Constraints

EXPRESS-G provides no methods for defining constraints, other than cardinalities. However, the fact that something is (elsewhere) constrained may be denoted by preceding the name of the thing with an asterisk (*) symbol.

- If an entity is a parameter in an EXPRESS rule, then the name of the entity may be preceded by an asterisk.

- If an attribute of an entity is constrained by either a UNIQUE clause or a WHERE clause within the entity, then the name of the attribute may be preceded by an asterisk.
B.5.4 Type modeling

EXPRESS-G provides the dashed box symbols (Figure B 4) for type definitions. The name of the definition is enclosed by the box.

A defined data type shall be modeled by the type definition symbol, the representation type definition, and the relationship line from the type definition to its attribute definition; the cardinality of the attribute may be placed on the relation.

NOTE 20 - An example of defined data type modeling can be seen in the strings type in Figure B 10.

A select type shall be modeled in a similar fashion to the defined data type, except that there will be one relationship and attribute definition for each of the selectable items. No cardinality shall be specified on the relations.

An enumeration type is modeled solely by its symbol. EXPRESS-G does not provide a mechanism for noting the enumerated items.

B.5.5 Entity modeling

EXPRESS-G provides the solid box symbol (Figure B 5) for ENTITY definitions. The name of the entity is enclosed by the box.

In EXPRESS, an entity may be part of a Supertype Tree, it may have Explicit attributes, it may have Derived attributes, it may have Inverse attributes, and attribute values may be constrained via UNIQUE and/or WHERE clauses.

EXPRESS-G supports Supertyping and Explicit, Derived, and Inverse attributes only. Each Explicit or Derived attribute in an EXPRESS entity gives rise to a relationship in the corresponding EXPRESS-G model. The role name of the attribute may placed on the relationship line, together with the cardinality which follows the role name. A Derived attribute shall be distinguished from an Explicit attribute by preceeding the name of the attribute by the characters DER enclosed in parentheses (i.e. "(DER)"). In the case where there is an Inverse attribute defined for an entity-entity relationship, the name and cardinality of the attribute shall be placed on the other side of the relationship line from the "forward" attribute, and the name shall be preceeded by the characters INV enclosed in parentheses (i.e. "(INV)").

In the cases where attribute values are constrained by UNIQUE or WHERE clauses in the EXPRESS model, the role name on the relation may be prefixed with an asterisk. Similarly, if an entity is constrained by a RULE in an EXPRESS model, then the name of the ENTITY may be prefixed with an asterisk.

NOTES

21 Typical Entity models can be seen in Figure B 1 and Figure B 10.

22 The indication of WHERE constraints on attributes can be seen on the roles husband and maiden-name in Figure B 1.

23 An example of Entities constrained by Rules is shown for the male and female entities in Figure B 1.

The Entities forming a Supertype Tree are connected by thick solid lines. The circled end of the relationship line denotes the Subtype end of the relationship. When a Supertype is ABSTRACT
the characters ABS, enclosed by parentheses, shall precede the name of the entity within the
entity symbol box.

**EXPRESS-G** provides only a limited notation for indicating the logical structure of a Supertype
tree. The oneof relation may be indicated by a "T branching" relationship line from the
Supertype to each of its Subtypes that are in a oneof relation to each other, together with the
digit "1" being placed at the T junction. No implications about the logical structure shall be
drawn from a T junction that is not marked by the digit 1.

![Supertype Tree Diagram](image)

Figure B 11 – Complete entity-level model of the Supertype tree from Example 122. (Page 1 of
1)

**NOTES**

24 Figure B 11, which is an EXPRESS-G model of Example 122, shows sub2 as being an
Abstract Supertype.

25 The model in Figure B 11 shows that the entities sub1, sub2, and sub5 are Subtypes of
the Supertype super. No implications can be made about the logical relationships between
these Subtypes (i.e., they may be in OR and/or AND relationship to each other). An instance
of super possibly has no Subtypes because it is not abstract. The entities sub3 and sub4
are Subtypes of the Supertype sub2. The entities sub3 and sub4 are in a oneof relation to
each other.

**B.5.6 Inter-schema references**

When a definition in one schema is utilized as part of a definition in another schema, this shall
be represented by the oval box containing the qualified name of the definition.

**NOTE 26**

Figure B 12 shows an entity-level model of a single schema. The EXPRESS source from
which this model was derived is given as Example 123.

The complete model consists of two schemas — top and geom (see Figure B 13) — and some
of the top schema entities have attributes that utilise definitions in the geom schema. As an
entity-level model only consists of those things defined in a single schema, the representation
of the top schema in this example requires the inter-schema references shown in the oval
boxes.
Figure B 12 – Complete entity-level model of the top schema of Example 123 illustrating inter-schema references. (Page 1 of 1).

B.6 Schema level model

The previous clause discussed the EXPRESS-G representations of a single schema model. This clause describes the representation of models consisting of multiple schemas.

The contents of an EXPRESS-G Schema representation are limited to noting the schemas comprising the model and the schema-schema relations. These are:

- The Schemas that another Schema USEs.
- The Schemas that another Schema REFERENCES.
- The names of the things that are referenced or used.

The USE relations shall be shown by a normal width relation line from the using schema to the used schema, with an open circle denoting the used schema end of the relationship line. The REFERENCE relations shall be shown by a dashed relation line from the referencing schema to the referenced schema, with an open circle denoting the referenced schema end of the relationship line.

The definitions that are used or referenced may be shown as a list of names adjacent to the relevant relationship line, and connected to the relationship line by an arrowheaded line. If a definition is aliased, then this may be indicated by following the original name of the definition by an equals (=) sign and the alias name.

NOTE 27 – A simple Schema model is shown in Figure B 13. This shows that there are
two Schemas, namely top and geom. The top schema Uses the geom schema. In particular, it references the surface and uses the curve and point definitions from the geom schema. The point definition is given the node alias in the top schema.

If a schema level model extends over more than one page and the schema-schema relationships cross the page boundaries, then the page referencing symbols shall be used.

NOTE 28 – Example 124 gives the EXPRESS source code for an abbreviated version of a multi-schema model. The EXPRESS-G schema model for this example is shown in Figure B 14.
B.7 Complete models

In EXPRESS-G a complete model is one that, within the limits of the EXPRESS-G notation, accurately represents all the relationships and constraints pertinent to a model — either an Entity-level model or a Schema-level model.

B.7.1 Complete entity-level model

The following requirements shall be satisfied by a Complete Entity-level model.

a) Each page of the model shall be titled, with the title starting with the words: “Complete entity-level model . . . ”.

b) Each page of the model shall be numbered in the form “Page X of N”, where N is the total number of pages forming the model, and X is the particular page number.

c) All entities, types, and simple type symbols used within a single schema shall be displayed.

d) Schema symbols shall not appear.

e) All relationships, role names, and cardinalities shall be displayed (i.e., attribute relations, including explicit, derived and inverse attributes, and Supertype relations).

f) All entity-entity relations that are not marked with an Inverse attribute shall be interpreted to have a cardinality of zero or more (i.e. there can be zero or more instances of the from entity with respect to one instance of the to entity).

g) All ABSTRACT Supertypes shall be marked.

h) All ONEOF Subtype relations shall be marked.

i) No logical structuring shall be imputed to an unmarked Subtype relation, except that it is not a ONEOF relation.

j) All definitions that are used or referenced from another schema shall be denoted by the oval box inter-schema composition symbols, together with the surrounding rectangular boxes of the appropriate style (i.e., solid for used definitions and dashed for referenced definitions). Any aliased names shall be noted in the relevant inter-schema reference symbol.

k) All entities that are constrained by a RULE shall be marked with an asterisk (*).

l) All attributes that are constrained shall be marked with an asterisk (*).

B.7.2 Complete schema-level model

The following requirements shall be satisfied by a Complete Schema-level model.

a) Each page of the model shall be titled, with the title starting with the words: “Complete schema-level model . . . ”.

b) Each page of the model shall be numbered in the form “Page X of N”, where N is the total number of pages forming the model, and X is the particular page number.

c) All schemas used within the model shall be displayed.

d) Neither entity nor type nor simple symbols shall be displayed.
e) All schema-schema relationships (i.e Use and Reference) shall be displayed.

f) The names of all definitions that are either used or referenced shall be attached to the relevant relationship line, together with any aliases. If no names are attached to a relationship line, this shall be interpreted to mean that the entire schema is used or referenced.
B.8 Partial models

When developing a model, or sometimes when viewing a model, it is useful to be able to display a model at varying levels of abstraction. The foregoing discussion was concerned with the representation of fully detailed models. EXPRESS-G provides some capabilities for the representation of an abstraction of an information model. An EXPRESS-G model that is not Complete is termed a Partial Model.

NOTE 29 — At this point it should be noted that an EXPRESS-G model is an abstraction of an EXPRESS model as the graphics notation only provides a subset of the semantic modeling capabilities of the EXPRESS language. Further, EXPRESS-G naturally provides two levels of abstraction in that a schema-level model displays none of the internal details of the schemas, which, if required, have to be displayed via individual entity-level models. There is no requirement in EXPRESS that an Entity must have attributes. In other cases, it may be known that an Entity will have attributes, but the number and type of the attributes are as yet undecided, or, for abstraction purposes, they are “minor” details that do not need to be displayed.

EXAMPLE 119 — A possible example of an unattributed entity could be a Vertex which conveys the concept of “location in some space” but without specifying an actual physical location. Thus

ENTITY vertex;
END_ENTITY;

would be a complete definition.

For a fully detailed EXPRESS-G model, every definition must be related to its “components” (i.e. an EXPRESS-G terminal or another definition). Depending on either the state of a model or a desire to eliminate minor details from a model, it is permissible in a Partial model to indicate the existence of a relationship from a definition yet not define the thing being related to. A relation line may end in a circle with no adjoining definition symbol box. The meaning of such a relation is unspecified (except insofar that it indicates that the model is incomplete), but it can be taken to imply one of the following:

- Further details are known to the producer of the diagram but are not exhibited.
- Further details are required but these were unknown at the time the diagram was produced.

![Diagram of a partial entity-level model showing an abstraction of a Person entity.](image)

Figure B.15 – Partial entity-level model showing an abstraction of a Person entity. (Page 1 of 1)
NOTE 30 — Figure B 15 shows an abstraction of the person entity from Example 120. The abstraction basically says that there is a thing called person which has some (undefined or undisplayed) attributes and which has some (undefined or undisplayed) Subtypes. Depending on the information modeling being performed, this abstraction could be the first attempt at a definition of a person, in which case the details will have to be developed. Alternately, it could appear in a much larger model where the concept of person was necessary, but of minor importance, to the overall model concept, and hence the person details were available but not displayed.

![Diagram](image)

Figure B 16 — Partial schema-level model of Example 124. (Page 1 of 1)

Just as with Entities, Schemas may also be abstracted using the same mechanism. Figure B 16 shows an abstraction of the schema-level model shown in full in Figure B 14. Here, the REFERENCE relations have been left unspecified.

B.8.1 Partial entity-level model

The following requirements shall be satisfied by a Partial Entity-level model.

a) Each page of the model shall be titled, with the title starting with the words: "Partial entity-level model ...".

b) Each page of the model shall be numbered in the form "Page X of N", where N is the total number of pages forming the model, and X is the particular page number.

c) Some or all of the entities, types, and simple type symbols used within a single schema may be displayed.

d) Schema symbols shall not appear.

e) Relationships and cardinalities may be displayed (i.e attribute relations, including explicit, derived, and inverse attributes, and Supertype relations).

f) Relationship lines may be ended by an open circle with no attached definition.

g) ABSTRACT Supertypes may be marked.

h) ONEOF relations among Subtypes may be marked. No interpretation shall be attributed to an unmarked Subtype relationship.
i) The oval box inter-schema reference symbols may be enclosed by the appropriately styled rectangular boxes, and alias names may be displayed within these boxes.

j) Entities that are constrained by a rule may be marked with an asterisk (*).

k) Attributes that are constrained may be marked with an asterisk (*).

B.8.2 Partial Schema-level Model

The following requirements shall be satisfied by a Partial Schema-level model.

a) Each page of the model shall be titled, with the title starting with the words: “Partial schema-level model . . .”.

b) Each page of the model shall be numbered in the form “Page X of N”, where N is the total number of pages forming the model, and X is the particular page number.

c) Some or all of the schemas used within the model may be displayed.

d) Neither entity nor type nor simple symbols shall be displayed.

e) Schema-schema relationships (i.e. USE and REFERENCE) may be displayed.

f) The names of all definitions that are either used or referenced may be attached to the relevant relationship line, together with any aliases. No interpretation of the meaning of a relationship line lacking definition names shall be made, except that of USE or REFERENCE, according to the line style. That is, the lack of names does not imply the USE or REFERENCE of an entire schema.
**Vi Quick Reference**

### Entering and Leaving vi

- `% vi name`  
  - edit name at top
- `% vi +n name`  
  - ... at line n
- `% vi + name`  
  - ... at end
- `% vi -r`  
  - list saved files
- `% vi -r name`  
  - recover file name
- `% vi name ...`  
  - edit first; rest via :n
- `% vi -1 tag`  
  - start at tag
- `% vi +pat name`  
  - search for pat
- `% view name`  
  - read only mode
- `ZZ`  
  - exit from vi, saving changes
- `CTRL-Z`  
  - stop vi for later resumption

### The Display

- **Last line**  
  - Error messages, echoing input to :/? and !, feedback about i/o and large changes.
- `@ lines`  
  - On screen only, not in file.
- `* lines`  
  - Lines past end of file.
- `CTRL-x`  
  - Control characters, DEL is delete.
- `tabs`  
  - Expand to spaces, cursor at last.

### Vi Modes

- **Command**  
  - Normal and initial state. Others return here. ESC (escape) cancels partial command.
- **Insert**  
  - Entered by `A` or `O` or `C` or `S` or `R`. Arbitrary text then terminates with ESC character, or abnormally with interrupt.
- **Last line**  
  - Reading input for :/? or ;; terminate with ESC or CR to execute, interrupt to cancel.

### Counts Before vi Commands

- line:column number
- scroll amount
- replicate amount
- simple insert
- repeat effect

### Simple Commands

- `dw`  
  - delete a word
- `de`  
  - ... leaving punctuation
- `dd`  
  - delete a line
- `3dd`  
  - ... 3 lines
- `k=:ESC`  
  - insert text above
- `cw`:ESC  
  - change word to new
- `xp`  
  - transpose characters

### Interrupting, Cancelling

- `ESC`  
  - end insert or incomplete cmd
- `CTRL-C`  
  - interrupt or DEL
- `CTRL-L`  
  - refresh screen if scrambled

### File Manipulation

- `:w`  
  - write back changes
- `:wq`  
  - write and quit
- `:q`  
  - quit
- `:q!`  
  - quit, discard changes
- `:e name`  
  - edit file name
- `:e!`  
  - edit, discard changes
- `:e + name`  
  - edit, starting at end
- `:e +n`  
  - edit starting at line n
- `:e #`  
  - edit alternate file
- `CTRL-=`  
  - synonym for `:e`  
- `:w name`  
  - write file name
- `:w! name`  
  - overwrite file name
- `:z`  
  - run shell, then return
- `:z cmd`  
  - run cmd, then return
- `:s`  
  - edit next file in arglist
- `:s args`  
  - specify new arglist
- `:s`  
  - show current file and line
- `CTRL-G`  
  - synonym for :Z
- `:ta tag`  
  - to tag file entry tag
- `CTRL-['  
  - :ta, following word is tag

### Positioning within File

- `CTRL-F`  
  - forward screenfull
- `CTRL-B`  
  - backward screenfull
- `CTRL-D`  
  - scroll down half screen
- `CTRL-U`  
  - scroll up half screen
- `G`  
  - go to line (end default)
- `?pat`  
  - next line matching pat
- `:pat n`  
  - prev line matching pat
- `N`  
  - reverse last / or ?
- `?pat+?n`  
  - n'th line after pat
- `?pat?n`  
  - n'th line before pat
- `:ll`  
  - next section/function
- `:ll`  
  - previous section/function
- `%`  
  - find matching () { or }

### Adjusting the Screen

- `CTRL-L`  
  - clear and redraw
- `CTRL-R`  
  - recype, eliminate @ lines
- `xCR`  
  - redraw, current line at window top
- `x`  
  - ... at bottom
- `x`  
  - ... at center
- `?pat=x`  
  - prev line at bottom
- `za`  
  - use a line window
- `CTRL-E`  
  - scroll window down 1 line
- `CTRL-Y`  
  - scroll window up 1 line

---

Sun Microsystems

Revision A. of 9 May 1982
Marking and Returning

..    previous context
..    ... at first non-white in line
mx    mark position with letter x
'x    to mark x
'x    ... at first non-white in line

Line Positioning

H    home window line
L    last window line
M    middle window line
+    next line, at first non-white
-    previous line, at first non-white
CR   return, same as +
\ or j  next line, same column
^ or k previous line, same column

Character Positioning

.    first non-blank
0    beginning of line
$    end of line
b or t  forward
l or l  backwards
CTRL-H same as +
space same as →
x    find x forward
Fx    f backward	x    upto x forward
Tx    back upto x
;    repeat last f F t or T
,    inverse of ;
(    to specified column
%    find matching ( ) or )

Words, Sentences, Paragraphs

w    word forward
b    back word
e    end of word
)    to next sentence
}    to next paragraph
(    back sentence
{    back paragraph
W    blank delimited word
B    back W
E    to end of W

Commands for LISP

)    Forward s-expression
}    ... but don't stop at atoms
(    Back s-expression
{    ... but don't stop at atoms

Corrections During Insert

CTRL-H erase last character
CTRL-W erases last word
erase your erase, same as CTRL-H
kill your kill, erase input this line
\ escapes CTRL-H, your erase and kill
ESC ends insertion, back to command
CTRL-C interrupt, terminates insert
CTRL-D backtab over autodent
CTRL-"D kill autodent, save for next
CTRL-D ... but at margin next also
CTRL-V quote non-printing character

Insert and Replace

a    append after cursor
I    insert before
A    append at end of line
I    insert before first non-blank
0    open line below
O    open above
rx   replace single char with x
R    replace characters

Operators (double to affect lines)

d    delete
c    change
<    left shift
>    right shift
!    filter through command
=    indent for LISP
y    yank lines to buffer

Miscellaneous Operations

C    change rest of line
D    delete rest of line
s    substitute chars
S    substitute lines
J    join lines
x    delete characters
X    ... before cursor
Y    yank lines

Yank and Put

p    put back lines
P    put before
"xp  put from buffer x
"xy  yank to buffer x
"x  delete into buffer x

Undo, Redo, Retrieve

u    undo last change
U    restore current line
.    repeat last change
*d p retrieve d-th last delete