U.S. DEPARTMENT OF COMMERCE
National Institute of Standards and Technology

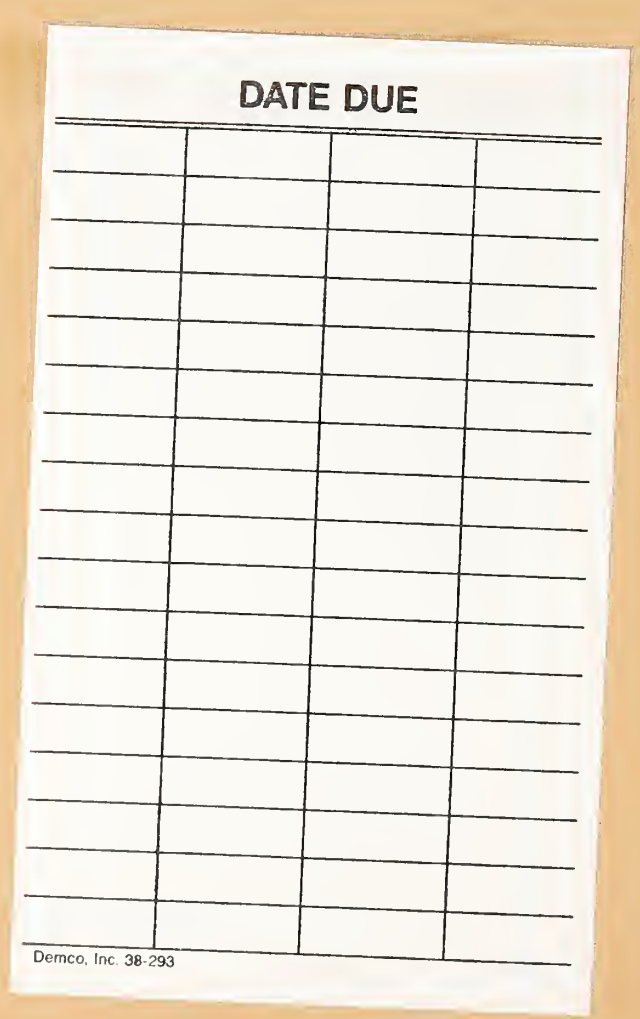
NISTIR 4361

National PDES Testbed Report Series

QDES User's Guide

NATIONAL PDES TESTBED

NIST



U.S. DEPARTMENT OF COMMERCE
National Institute of Standards and Technology

NISTIR 4361

National PDES Testbed

QDES User's Guide

Stephen Nowland Clark

U.S. DEPARTMENT OF COMMERCE
Robert A. Mosbacher,
Secretary of Commerce

National Institute of Standards and Technology
John W. Lyons, Director

June 29, 1990

NIST

Disclaimer

No approval or endorsement of any commercial product by the National Institute of Standards and Technology is intended or implied

Smalltalk-80 is a trademark of ParcPlace Systems, Inc.

Macintosh is a registered trademark of Apple Computer, Inc.

Table Of Contents

QDES User's Guide

Stephen Nowland Clark

1 Introduction

QDES, the Quick-and-Dirty Editor for STEP, is intended to provide a temporary facility for adding tolerancing and other data to a PDES geometric model. The expectation is that a solid modeller will be used to produce a STEP physical file. This file can then be imported into QDES, which provides a window-based structured editing environment to ease the task of adding such objects as tolerances and form features to the model.

QDES is an information-model-driven tool: The editor itself has no *a priori* knowledge of any particular information model. Instead, a set of Smalltalk classes are loaded into the QDES environment, and QDES interprets these classes to determine its domain. In the PDES world, this information model is specified in the Express language [Schenck89]. A translator based on the Fed-X Express parser [Clark90b] has been built at NIST to produce these Smalltalk classes from an Express information model.

Administrative procedures, such as importing a new schema or releasing QDES, are described in [Clark90d].

1.1 Context

The PDES (Product Data Exchange using STEP) activity is the United States' effort in support of the Standard for the Exchange of Product Model Data (STEP), an emerging international standard for the interchange of product data between various vendors' CAD/CAM systems and other manufacturing-related software [Smith88]. A National PDES Testbed has been established at the National Institute of Standards and Technology to provide testing and validation facilities for the emerging standard. The Testbed is funded by the CALS (Computer-aided Acquisition and Logistic Support) program of the Office of the Secretary of Defense. As part of the testing effort, NIST is charged with providing a software toolkit for manipulating PDES data. This NIST PDES Toolkit is an evolving, research-oriented set of software tools. This document is one of a set of reports which describe various aspects of the Toolkit. An overview of the Toolkit is provided in [Clark90a], along with references to the other documents in the set.

For further information on QDES or other components of the Toolkit, or to obtain a copy of the software, use the attached order form.

2 Terminology

In order to avoid later confusion, let's quickly define a few terms:

- Entity or Object

  An *entity* is a single object in the database, or a component of a product. In the PDES world, for example, the point (1.2, 3.4, 5.6), a planar face, and a flatness tolerance on that face are all entities.

- Class or Entity Class

  A *class* is a type of entity, corresponding to a definition in a schema. It is also useful to think of a class as a collection of entities sharing a common set of defining attributes. For example, `cartesian_point` and `topology` are entity classes; the former can be thought of as the collection of all points in cartesian space.

- Member

  A *member* of a class is an entity which is in the collection represented by that class. Thus, (1.2, 3.4, 5.6) is simultaneously a member of all of the classes `cartesian_point`, `point`, and `geometry`.

- Instance

  An *instance* of a class is a member of the class, but not of any of its subclasses. Every object is an instance of exactly one class, but may be a member of several classes. Thus, (1.2, 3.4, 5.6) is an instance of the class `cartesian_point`, but not of the class `point`. `Cartesian_point` is called the *class of* this object.

- Model, Database, or Workspace

  These words all refer to the set of all entity instances currentlyin the system.

- Schema

  The word *schema* will be used to refer to the entire conceptual schema (information model), which, in Express terminology, is actually composed of a number of subschemas.

3 The QDES Environment

QDES is implemented in the Smalltalk-80™ programming environment. This environment provides a set of user-interface tools (windows, menus, and structured views) in an object-oriented environment which is well suited to dealing with the hierarchical nature of an Express information model. Since Smalltalk-80 is a completely integrated programming environment, with extensive run-time support and user-interface tools, it has left a strong imprint on the look and feel of the editor.

QDES makes full use of the windowing system provided by Smalltalk-80. There are several types of windows which may appear at various points in a QDES session. These include Browsers, Hierarchy and Structure Graphs, a Feedback window, and Notifiers.

The Feedback window should always be present, as this is where QDES displays important, although not critical, feedback to the user. If this window ever disappears during a session (which it should not), it can be recreated by selecting "feedback window" from the system menu, described below.

Browsers and Graphs come and go at the whim of the user; these are the windows in which the actual work of the system is done. A Browser is a highly structured window which is used to view and edit the actual text of entity definitions. A Graph can be used to graphically view the class hierarchy defined in the Schema or the internal structure of an entity.

Notifiers appear when critical errors occur which leave the system unable to perform as expected. A Notifier appears, for example, when a type mismatch is detected in the value provided for an entity attribute, or when an attempt is made to type text into a read-only view. A Notifier has a thicker border and bolder text than other windows in the system; it is also distinguished by the fact that it temporarily takes complete control over the system: it will not allow QDES to proceed until the user acknowledges the error message by clicking any mouse button.

3.1 System Control

The intended mode of operation in QDES is to use the Hierarchy and Structure Graphs to navigate in the (very large) Schema and the (possibly very large) model, and then to create Browsers on the portions of interest. Thus, one might use a Hierarchy Graph to find the class `cartesian_point` in the schema, and then spawn a Browser on all instances of this class in the current model. Or, one might create a Structure Graph on some tolerance in the model, use this view to find the toleranced feature, and then spawn a Browser on that particular feature. In either scenario, the resulting Browser could be used to modify the entities found or to add new entities to the workspace. New Structure Graphs could also be spawned from these Browsers, allowing further navigation in the system.

3.1.1 Starting up a QDES Session

The command used to invoke QDES includes an indication of the conceptual schema which will be used for the session. Check locally to determine what models are available; typing "`qdes`" at the prompt may give a somewhat accurate list of your options. At this writing, three versions are available at NIST: `qdes-ipim` contains the full Tokyo IPIM[1], and two of PDES Inc.'s CDIMs[2] are in `qdes-cdima` and `qdes-cdimb`. There are no command-line options to QDES; a CDIM A session is initiated by typing

```
% qdes-cdima
```

3.1.2 Dealing with Menus and Windows

Since the Smalltalk-80 window system has its own unique interface paradigm and associated quirks, we'll take a look at how these are reflected in the QDES interface before proceeding to a description of the actual mechanics of getting around in the editor. Familiarity with mice and popup menus in general is assumed.

---

1. The Integrated Product Information Model is the integrated portion of the proposed PDES information model; see [Smith88].
2. These Context-Driven Information Models are described in [Mitchell90].

3.1.2.1 Walking Menus

Occasionally, a presented menu will have one or more items marked with an open triangle. This indicates a walking menu: moving the mouse to the right, past the triangle, pops up a submenu based on the marked item. Walking menus are used primarily for class selections, where the tree structure of the menu parallels the class hierarchy defined by the conceptual schema.

Another feature of menus which is encountered in the class hierarchy menu is scrollability. When a menu contains too many items to fit on the screen vertically, it becomes scrollable. There is no visual indication that a menu can scroll, but it is usually a good guess when a menu goes from (nearly) the top of the screen to (nearly) the bottom of the screen. To scroll such a menu, simply move the mouse past the top or bottom of the menu while holding down the mouse button which controls the menu.

3.1.2.2 Scrollbars

Smalltalk-80 scrollbars, which appear on most views in QDES, are unlike any other scrollbars in the world (probably). Each scrollbar is divided horizontally into three separate areas of sensitivity; the current area is indicated by the shape of the cursor: in the left area, the cursor is a down-arrow; in the center, a right-arrow; and in the right, an up-arrow. In the middle area a marker appears which shows the current position of the view in its (textual) model. The height of the marker is proportional to the fraction of the contents of the view which is visible.

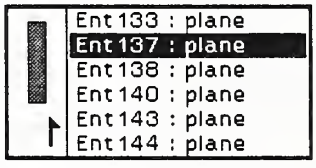


Figure 1: View With Scroll Bar

In each area of the scrollbar, all three mouse buttons perform the same actions. Clicking in the left-hand area brings the top line of text in the view down to the current cursor position (hence, the down-arrow indication). To repeat this action, hold down the button. Similarly, clicking in the right-hand area scrolls the line of text next to the cursor to the top of the view; this action is also repeated if the button is held down. To scroll to an absolute position in the view, click in the middle area. The marker will be centered around the cursor, and the view adjusted accordingly. As long as the mouse button is held down, the marker (and the view) will continue to follow vertical movements of the cursor.

3.1.2.3 Windows

Smalltalk indicates the currently active window (the window to which typing is directed) by inverting its title bar (white text in a black bar). The way in which control changes from one window to another is somewhat odd, and clearly deserves comment. The

current window can only receive keyboard input when the mouse cursor is within its borders. When the mouse is moved out of the current window, the window retains control but is not active. Keyboard input is queued, but is not directed to any window. Because the active window retains control, if the mouse is moved back into the same window, operation resumes with the same active window. In order to change to a different active window, the mouse must be moved into the new window and any button clicked. In either case, any queued keyboard input is immediately dumped into the (possibly new) active window.

Smalltalk-80 assigns consistent functionality to each of the three mouse buttons, and the same conventions are adopted by QDES. The left button is used to make selections throughout the system. It is used to select text (see section 4.1.3.3), to click on buttons, to select elements in Graph Views, and to choose options from static menus. The middle button has an associated popup menu which provides functions which are specific to the current location of the mouse. Thus, when the mouse is not in any window, pressing the middle button gives a menu of system control operations, such as "load file," "restore display," and "quit." In different panes of a Browser, the middle button has different menus attached to it. And in a Graph View, there is yet another menu of operations which can be performed on the current selection.

When the mouse is in a window, pressing the right button gives a menu of basic window operations. These operations are generic to all Smalltalk-80 windows, and are described below.

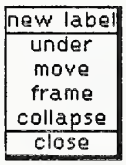


Figure 2: The Window Manipulation Menu

- New label: This is probably not terribly useful in the QDES environment, but it prompts for a new label for the current window. Just hitting <return> selects the default value, which is the current label.
- Under: This selection pushes the current window underneath any windows which it overlaps.
- Move: This option allows the current window to be moved. The upper left corner of the window is attached to the mouse, and follows it around until any button is clicked.
- Frame: This option allows the current window to be repositioned and resized. The window is first repositioned as in move. The mouse is then attached to the lower right corner of the window, allowing the window to be reshaped, until a button is clicked again.

- Collapse / Expand: When collapse is selected, the current window is shrunk to an icon (a small rectangle containing only the window's label). When a window is collapsed for the first time, this icon must be positioned by clicking the mouse as in move; this position is remembered in the future. "Collapse" is replaced with "expand" on the menu for the icon; selecting "expand" restores the window to its original grandeur.
- Close: This option destroys a window, removing it from the display. It has no effect on the contents of the window. In particular, closing a Browser does not change the entities browsed.

3.1.2.4 Special Windows

There are various kinds of special windows which are occasionally encountered in QDES. These windows often appear when QDES needs some specific piece of information before it is able to proceed; they thus take complete control of the system, not allowing QDES to proceed until the user performs some specific action. In these windows, the mouse buttons often do not function as usual; in some cases, the buttons do nothing at all.

One such window is the Notifier. As mentioned above, when some serious error condition occurs, a Notifier appears, and all processing halts until the user clicks any mouse button, at which point the Notifier disappears and QDES returns to business as usual. Examples of "serious error conditions" include such things as: a request for a non-existent file; a type clash in the user's input; and an attempt to delete an entity which is referenced by another entity.

A second special type of window is the Confirmer. This window is used to get an immediate yes-or-no answer to some very important question (for example, "Do you really want to quit QDES?"). A Confirmer displays a question to be answered and contains buttons marked "yes" and "no." QDES will not proceed until one of these responses is selected with the left button; neither of the other mouse buttons has any effect.

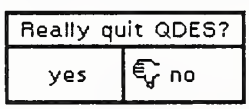


Figure 3: A Confirmer

The last special type of window is the Prompter. This window is used when an immediate textual reponse is needed, such as a file name for a save operation. It does not relinquish control until the user enters some text and hits <return>. In a Prompter, the left button can be used for text selection, and the middle button pops up a menu similar to the text manipulation menu (section 4.1.3.3).

3.1.3 The System Menu

The system menu, which is attached to the middle button when the mouse is not in any window, gives access to a number of basic system functions, such as file manipulation and the creation of Browser and Graph windows.

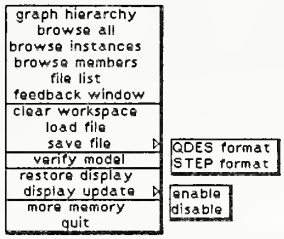


Figure 4: The System Menu

- Graph hierarchy: This menu item creates a Hierarchy Graph showing all of the entity types in the schema. This may take some time if the schema is large.
- Browse all: This menu item creates a System Browser, described below. The Browser contains the entire model. This window is the basic tool for editing and viewing the model.
- Browse instances: This menu item pops up a walking menu mirroring the class hierarchy of the conceptual schema. When a class is selected, a Browser is created which contains all of the instances, if any, of the selected class.
- Browse members: This menu item pops up a walking menu mirroring the class hierarchy of the conceptual schema. When a class is selected, a Browser is created which contains all of the members, if any, of the selected class.
- File list: This item opens a file list window, a standard Smalltalk-80 tool for browsing for files in the file hierarchy. The "file [it] in" option loads a QDES intermediate format file into the model editor. A QDES intermediate format file can be produced either by pushing a STEP physical file through STEPparse-QDES (see section 3.1.4) or by saving a model from QDES.
- Feedback window: Normally not used, this menu item creates a replacement for an accidentally lost Feedback window.
- Clear workspace: Selecting this menu deletes the entire model from memory, so one can start from scratch. Fortunately, QDES asks for confirmation before doing this. The conceptual schema remains untouched.
- Load file: The load file option instructs QDES to read a set of entities from a QDES intermediate format file into the current model. A Prompter requests the name of the file to be loaded.

- Save file: QDES can write out a model in either of two formats. The "QDES format" option of this menu writes out a QDES intermediate file (which can be directly loaded back into QDES). The "STEP format" option produces a STEP physical file [Altemueller88]. In either case, a Prompter requests the file name to use; if the named file exists, QDES asks confirmation before over-writing it.
- Verify model: Although QDES will happily write most inconsistent or incomplete models to disk, this menu item allows a user to check the integrity of a model before doing so. This function currently reports errors such as missing non-optional attributes and apparent circularities in the data structures. In the future, Express `where` clauses and other constraints might also be checked.
- Restore display: Redisplay all QDES windows, for damage repair.
- Display update: This item holds a submenu with two items: "enable" and "disable." These can be used to enable and disable the automatic update of the display after changes to the model. For example, loading a file takes significantly less time with updating disabled; a single update can be done at the end of the load instead. Selecting the "display update" option itself causes QDES to show in the Feedback window the current status of automatic display updating.
- Quit: After asking for confirmation, quit QDES. Any modeling changes which have been made are lost, unless the model has been saved as a QDES intermediate file or a STEP file.

3.1.4 Getting a STEP Model into QDES

It is not intended that QDES be used to create the geometry and topology of a product model. This process is several orders of magnitude easier with a geometric modelling system than with a low-level text-based system such as QDES. The intention is rather that a geometric model first be created and stored as a STEP physical file. QDES can then be used, for example, to add tolerancing information to the model.

In order to be loaded into QDES, a STEP physical file must first be translated into a special QDES file. This is accomplished by pushing the file through the STEPparse-QDES translator:

```
% stepparse_qdes -e <express> -s <STEP>
output file: <file>.st
```

Where `<STEP>` is the STEP file to be translated and `<express>` is the Express conceptual schema used by this model. The resulting QDES intermediate file is written to `<file>.st`.

Once a QDES file has been produced, it can be loaded into the editor by simply selecting the "load file" option from the system menu and typing the name of the QDES file. As mentioned above, this operation experiences a *significant* (read: exponential) performance increase when automatic display updating is temporarily disabled. Also, note that the load operation does not clear the workspace before reading new data. This must be done explicitly in order to maintain the integrity of separate models.

3.2 Browsers

So there's a Browser on the screen, and a model in the workspace. Now what? What can I do with it? In this section, we get a grip on just what a Browser might be.

A Browser is the primary vehicle for creating, modifying, and viewing entities in a PDES/STEP model. The Browsers actually provide all of the functionality needed to view or edit a model in a structured manner. There are two primary types of Browsers: those which are attached to a single entity in the model (Entity Browsers), and those which can contain any number of entities (System and Filtered Browsers).

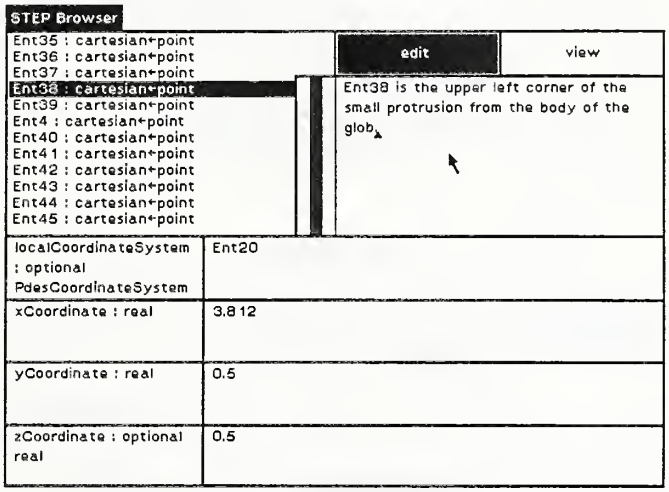


Figure 5: A System Browser

The first Browser encountered will probably be a System Browser, as this can be created from the root menu. This Browser provides access to all the entities in the system. It has several panes: a selection pane, which lists all of the entities in the system, and in which entities can be selected for manipulation; a comment pane, which shows a comment associated with the currently selected entity; a form pane, which shows the meat of the selected entity's definition; and a pair of buttons for selecting view (read-only) or edit (read-write) mode for the Browser.

A Filtered Browser looks just like a System Browser, but its selection pane is filtered by some rule. Thus, we can have a Browser showing all entities which are instances of some class, or those which are members of some collection.

An Entity Browser looks somewhat different: being permanently attached to a single entity, it has no need for a selection pane, and is suitably rearranged. With this exception, an Entity Browser is functionally very similar to the other two.

The form pane is effectively a form with blanks showing the various attributes of the selected entity and their values. In edit mode, these blanks can be filled in or changed. When a blank contains structured data, the data is abbreviated to fit in the space al-

lowed, and the user can request another Browser to examine this structured data. If, for example, a particular attribute of some entity being browsed contains another entity, only the name of the contained entity is shown; an Entity Browser may be requested if more detail is needed.

3.3 Graph Views

The two kinds of Graph views are very similar in appearance, although their respective middle-button menus are quite different, owing to the different types of structures which they view. A Graph view is composed of nodes (rectangles containing text, representing actual entities or classes) and arcs (arrows indicating relationships). In a Hierarchy Graph, the nodes correspond to entity classes, and the arcs represent the subtype relationship: every class points to each of its subtypes. This provides a vivid illustration of the relationships among the various entity classes in the schema. In a Structure Graph, the nodes correspond to entities and primitive objects (numbers, logicals, etc.), while the relationship represented by the arcs is containment. Thus, each entity points to all of its components, again providing a vivid illustration of the internal structure of an entity.

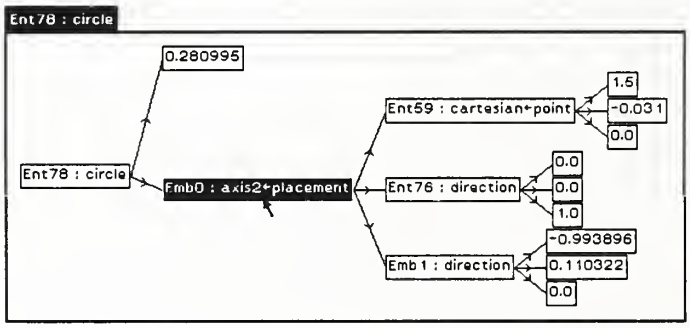


Figure 6: An Entity Graph

4 The Browsers

OK. Now it's time to get down to the nitty-gritty. What buttons and keys do I press when, and what does all that stuff on the screen really mean?

4.1 System and Filtered Browsers

Since nearly all Browser features appear in the multi-entity version, we will discuss this one first. As mentioned above, the System and Filtered Browsers are virtually indistinguishable; in fact, there is no need for the user even to know that two different kinds exist.

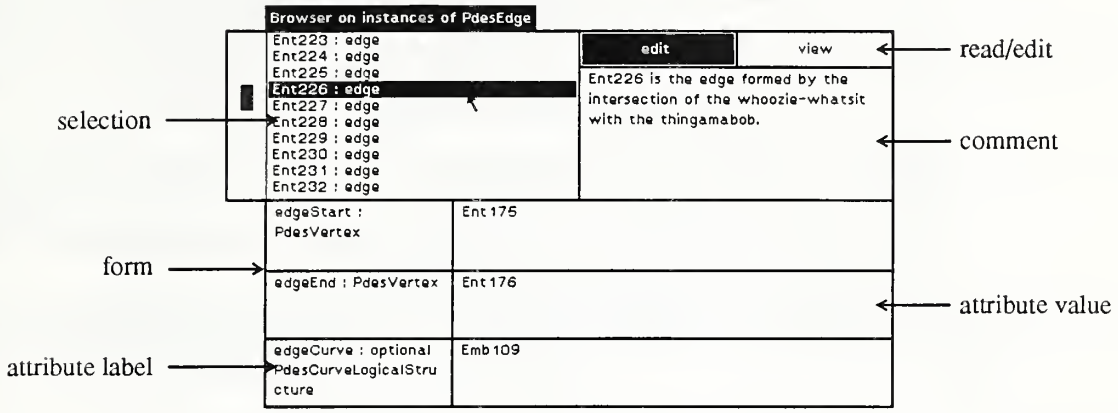


Figure 7: A Filtered Browser

4.1.1 View/Edit Toggle

Every Browser contains two buttons in the upper right corner. These buttons, marked "edit" and "view," provide some security against accidental modification of the model. Exactly one of these buttons is selected (marked with a solid black rectangle) in each Browser at all times. Selecting one with the left mouse button deselects the other. The model cannot be modified from a Browser marked "view" (or read-only). Browsers started from the system menu, as well as those spawned by Graph views and read-only Browsers, are initially marked "view," although a user may immediately select "edit."

4.1.2 Selection Pane: Which Entity Am I Editing?

The selection pane gives the highest level of control over a Browser. It lists all of the entities accessible through the Browser, together with the class of each. Clicking on the name of an entity with the left button selects it as the active entity; clicking on the active entity deselects it, so that the Browser has no active entity. The active entity is highlighted with a black bar in this pane. Since the Browser will often contain a large number of entities, this pane is scrollable.

The middle-button gives two different menus in the selection pane, depending on whether or not there is an active entity. The menu when there is no selection contains the following options:

- Update browser: Some changes to the model are not correctly propagated to all affected Browsers. This selection tells the Browser to "look again," and to make any needed adjustments to itself.
- Find entity: The Browser prompts for a name and, if an entity with the name exists, makes it the current selection.

- Create entity: A walking menu is presented from which to select the class of a new entity. The operation can be aborted at this point by clicking outside of the menu. After this selection is made, QDES requests a name for the new entity, which is then created. If no name is entered, a new, unique symbol is generated and used as the name of the entity. (This item generates an error in a read-only Browser)
- Paste: If the clipboard (see copy entity, below) currently holds an entity, QDES prompts for a new name and creates a carbon copy of the clipboard contents under the new name. Again, if no name is entered, a unique name will be generated. (This item generates an error in a read-only Browser)

When there is an active selection, the middle-button menu in the selection pane includes all of the above options, as well as several other options for manipulating the selected entity:

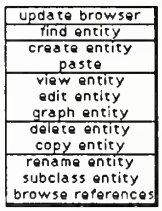


Figure 8: The Full Selection Pane Menu

- View entity: Spawns a read-only Entity Browser on the active entity.
- Edit entity: Spawns an editable Entity Browser on the currently selected entity. Note that a read-only Browser cannot spawn an editable Browser; in this situation, this option is equivalent to view entity.
- Graph entity: Creates a Structure Graph on the active entity.
- Delete entity: Deletes the active entity from the model. This operation does not deal properly with any Entity Browsers which might be attached to the deleted instance. This is a serious problem. Before deleting an entity instance from a System or Filtered Browser, be sure to close any Entity Browsers attached to the instance.
- Copy entity: This menu item places the current entity on the QDES clipboard, a conceptual holding area for entities being duplicated. The entity can later be retrieved and a duplicate created (see paste, above); it can also be pasted into an appropriate attribute of some other entity (see paste in section 4.1.3.1).
- Rename entity: QDES prompts for a new name for the active entity. The default value, selected simply by hitting <return>, is the entity's current name.

- Subclass entity: Presents a walking menu reflecting the portion of the class hierarchy rooted at the class of the current entity. When a class is selected from this, the current entity becomes an instance of the new subclass. All references are preserved (and remain consistent, due to the nature of the subclass relationship); the entity simply gains the attributes of the new subclass.
- Browse references: Spawns a Filtered Browser which contains all entities of which the current entity is a component.

4.1.3 Form Pane: Where the Real Work Happens

The lower two-thirds or so of any Browser consists of the form pane. This pane displays the definition of the active entity (if there is one) and, in an editable Browser, allows the user to modify this definition.

The form pane is in turn composed of a number of field or attribute views, each corresponding to a single attribute in the definition of the entity. Thus, if the current selection is a `cartesian_point`, the form pane will contain four fields: one for a `local_coordinate_system`, and three for the x, y, and (optional) z coordinates. On the left side of each field is a label which shows the name of the corresponding attribute and its type; the type is preceded by the word "optional" if the attribute is defined to be optional. The remainder of the field is a scrollable text area containing the value of the attribute.

4.1.3.1 The Field Menu: Manipulating Values

Each field label has an associated middle-button menu which provides functions for manipulating the value of the attribute. There are three possible menus, depending on the attribute type: entity, aggregate, or simple type.

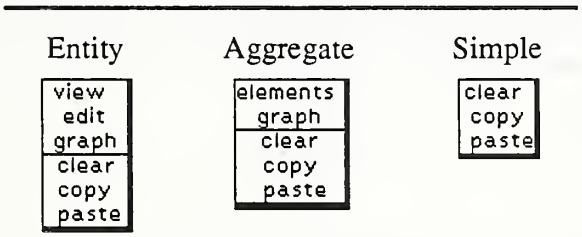


Figure 9: The Attribute Label Menus

For an attribute which is itself an entity, there are three special menu options:

- View: Spawns a read-only Entity Browser on the value of the attribute (no action is taken if the field is empty).
- Edit: Spawns an editable Entity Browser on the value of the attribute. If the current Browser is read-only, this item behaves exactly like View. Otherwise, if the field is empty, a new entity is created and placed in the field. A Prompter prompts for a name for the new entity; if the name already exists in the model, QDES attempts to use the named entity. Otherwise, a new, empty entity of the appropriate class is cre-

ated. If a more specific entity is required (e.g., a `cartesian_point` in an attribute which expects a `point`), the new entity can be subclassed by selecting the "subclass" option from the selection pane menu.

- Graph: Spawns an Entity Graph View of the value of the attribute.

For an attribute which is a collection (an aggregate, in Express terminology), there are two special options:

- Elements: Spawns a Filtered Browser which contains all entities in the attribute's value.
- Graph: Spawns an Entity Graph View of the value of the attribute.

In addition, all three field menus share three options:

- Clear: Clear (nullify) the attribute.
- Copy: Copy the value of the attribute to the QDES clipboard.
- Paste: If the clipboard is not empty, try to paste its contents into the attribute. This operation fails, and a Notifier appears, if the type of the clipboard contents is not appropriate for the attribute in question.

4.1.3.2 Textual Representation of Values

Simple types in QDES are represented in predictable fashion: Numbers as digit strings (scientific notation is allowed for reals); logical values as `true` or `.T.` and `false` or `.F.`

, enumeration values as character strings, and strings between single quotes (note that a string which contains no white space can be entered by the user without the quotes; for any string with white space, however, the quote marks must be entered). An entity is represented by its name. An aggregate (array, list, or set) is represented by a series of values of the appropriate type, separated by white space, and enclosed in parentheses.

- Examples:

  integer: `34`

  boolean: `true` or `.F.`

  enumeration: `wavy_line`

  string: `'this is a string'` or `nospaces`

  entity: `ent35`

  list of entities: `(ent34 ent35 ent36)`

Values of Express `select` types are handled differently. When the type of an attribute is a `select` type, the corresponding value pane is further subdivided, as if it were a form pane. Each field pane corresponds to a possible type for the attribute. QDES ensures that only one field is filled in at any given time.

4.1.3.3 Manipulating Text

The text selection mechanisms provided by Smalltalk-80 are similar to those available on an Apple Macintosh®. The insertion point is marked by a subscript caret. Clicking the left mouse button at some position in the text moves the insertion point to the indicated location. Holding down the left button while moving the mouse (click-and-drag) selects all of the text traversed. Selected text is displayed in inverse video.

Double-clicking (clicking two consecutive times at the same location) has several possible results, depending on the textual context of the selection. Normally, double-clicking selects the entire word surrounding the picked location. Double-clicking immediately inside a delimiter (such as a parenthesis or single- or double-quote) selects the entire body of text between the delimiter and its companion. Double-clicking past the end of a line of text selects the entire line; and past the end of a paragraph (eg, the value of an attribute), the entire paragraph. In all of these cases, a third click in the same location undoes the effect of the selection. Note that double-clicking in Smalltalk is slightly different from other styles: the delay between two clicks is not significant. A double-click is determined only by the spacial distance between two consecutive clicks, and not by the temporal distance.

Selected text can be manipuated in several ways. If anything is typed on the keyboard, the selection is deleted and replaced with the new text. In addition, the middle button menu in an attribute value text pane provides copy, cut, and paste functions which operate on the text selection (if any) in the pane. Note that these do not use the same clipboard as the copy and paste operations on the selection and label panes. The text copy, cut, and paste operations use a more primitive clipboard which simply holds text. Thus, the operations do no type checking: it is perfectly legal to copy a string "foo" from one pane and paste it into a different pane which expects an integer value.

Figure 10: The Text Editing Menu

Text which is typed into QDES (in a Prompter or an attribute value pane) is not "seen" by the editor until the user hits <return> (or chooses "accept" from the text menu, which has the same effect and takes more time to do). At this point, the text is sent to QDES, which (in the case of an attribute value) attempts to parse the appropriate type of value from the input string. If the expected type is an entity, and if a name is entered which does not refer to an existing entity, a new entity of the appropriate type is created. Similarly, any undefined names in an aggregate of entity names are created as new entities. When the base type of an aggregate is an Express `select` type, however, all names must previously have been defined as one of the expected types. It is currently not possible for QDES to create an appropriate object "on the fly" in this case.

QDES provides a limited undo facility for textual changes: at any time, any textual pane can be backed out to its last accepted value by choosing "revert" from the middle button menu. Note that this means that, if an error is discovered after the erroneous value has been accepted, all hope is lost (!).

4.1.4 Comment Pane: What's Ent37?

Every Browser includes a pane which contains a comment describing the currently active entity. This comment can be edited as if it were an attribute value; unlike the attribute value panes, however, the comment pane accepts <return> as a character, rather than as a command. Thus, QDES reads the new comment only when "accept" is chosen from the middle-button menu.

It is important to note that the STEP file format makes no provision for formally attaching comments to entities. These comments are saved only when a model is written in QDES format; they are lost when a STEP file is written.

4.1.5 A Note on the Consistency of Filtered Browsers

Finally, a note on Filtered Browsers and how accurately they reflect reality. In general, changes such as adding new instances to a class or deleting elements from a collection often are not correctly propagated to affected Filtered Browsers. When changes of this nature are made, it is best to select "update" from the selection pane middle-button menu in any potentially affected Filtered Browsers. This will ensure that these Browsers reflect their models accurately at all times.

4.2 Entity Browser

An Entity Browser, which is attached to exactly one entity for its entire lifetime, is both similar to and different from a multi-entity Browser in predictable ways.

4.2.1 How it's Like a System Browser

An Entity Browser has all of the same panes as a System Browser, with the exception of the selection pane, which is no longer necessary. These panes function in exactly the same way as their System Browser counterparts.

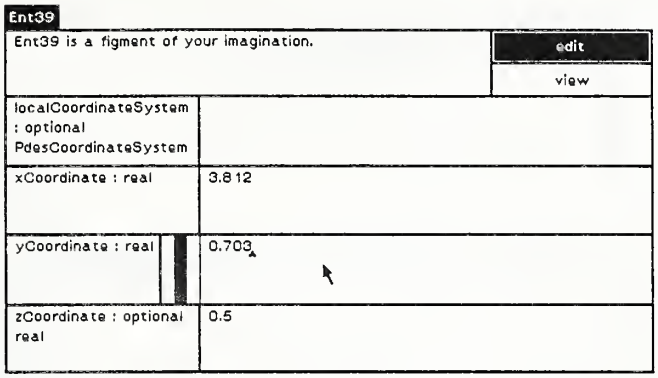


Figure 11: An Entity Browser

4.2.2 How it's Different from a System Browser

Instead of a selection pane, an Entity Browser uses its label to show the name of the browsed entity. This label has an asssociated middle-button menu which provides the relevant selections from the missing selection pane menu. The menu options which are available are: graph, delete, copy, rename, and subclass entity; and browse references. These are identical to the corresponding functions on the selection pane menu.

5 The Hierarchy Graph

It is sometimes useful to be able to view the class hierarchy (or some portion of it) defined by the conceptual schema, and to manipulate the various classes. This is the purpose of the Hierarchy Graph. The class hierarchy is shown as a tree structure (in reality, the structure is a directed acyclic graph, since the schema may make use of multiple inheritance) which grows from left to right, and which is rooted at a node called "PDES." Each entity class is represented by a node, which appears as a box containing the name of the class. Arrows connect each class to each of its subclasses.

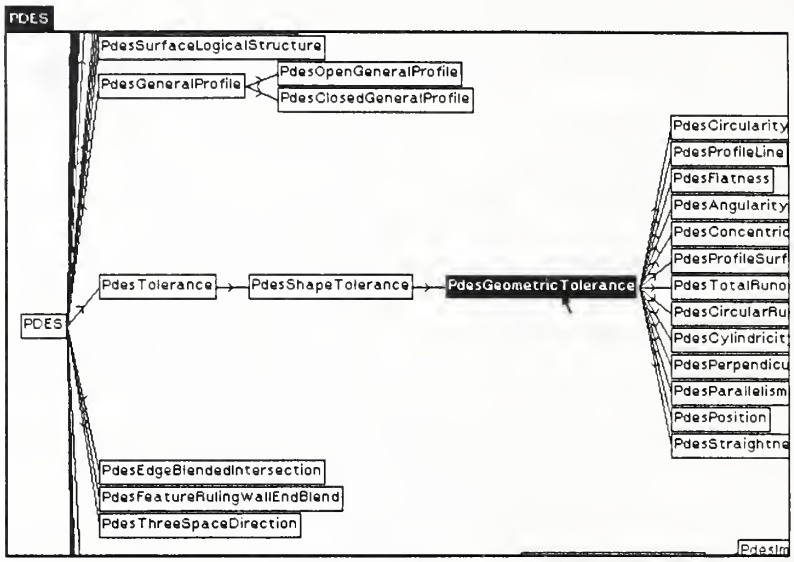


Figure 12: Portion of a Class Hierarchy Graph

The class nodes in the graph can be selected by clicking with the left mouse button. Selecting a class deselects the previously selected class; and clicking on the current selection causes it to be deselected. When a node is selected, its box is displayed in reverse video.

5.1 What Can I Do With My Selection?

The middle-button menu in a hierarchy graph provides functions for manipulating the selected class.

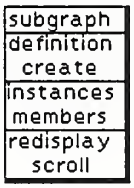


Figure 13: The Hierarchy Graph Menu

- Subgraph: Spawns a Hierarchy Graph on that portion of the class hierarchy rooted at the currently selected node.
- Definition: Primarily of use to QDES programmers, this item spawns a Smalltalk-80 Inspector on the selected Express entity class.

- Create: Prompts for an entity name and creates a new instance of the selected class with the given name. As with other entity creation operations, if an entity with the given name already exists, QDES will use it if it is of an appropriate type; otherwise, an error is reported via a Notifier, and the operation has no effect.
- Instances: Spawns a Filtered Browser showing all instances of the selected class.
- Members: Spawns a Filtered Browser showing all members of the selected class; i.e., all instances of the selected class, together with all instances of all of its descendants.
- Redisplay: Redisplays the contents of the graph view. This is needed, for example, after resizing the window. This item does not require a current selection.
- Scroll: Displays a reduced image of the entire class hierarchy in the upper left corner of the view. A box representing the current position of the window can be dragged by holding down the left mouse button while moving the mouse. This item does not require a current selection.

Note that the first five items have no effect if there is no current selection. This is indicated by causing the graph view to flash once.

6 The Entity Graph

Similar to the Class Hierarchy Graph is the Entity Graph. In this view, the nodes of the graph represent actual values in the workspace: entities, aggregates, integers, etc. The arrows represent inclusion. Thus, arrows connect each entity to its attributes, and each aggregate to its constituents.

Primitive objects (numbers, strings, and booleans) are represented simply by their values. The label for an aggregate is "Set," "Array," or "OrderedCollection," the latter representing a PDES list. The label for an entity consists of the name of the instance and the name of the entity's class.

6.1 What Can I Do With My Selection?

As in the case of the Hierarchy Graph, several items on the Entity Graph middle-button menu cause the view to flash, indicating that no action was taken, when there is no currently selected object.

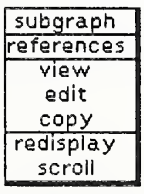


Figure 14: The Entity Graph Menu

- Subgraph: Spawns an Entity Graph on the currently selected object.
- View: Spawns a read-only Entity Browser on the currently selected entity. If the current selection is not an entity, the view flashes and no action is taken.
- Edit: Spawns an editable Entity Browser on the currently selected entity. If the current selection is not an entity, the view flashes and no action is taken.
- Copy: Copies the current selection to the clipboard.
- Redisplay: Redisplays the contents of the Graph View. This is needed, for example, after resizing the window. This item does not require a current selection.
- Scroll: Displays a reduced image of the entire class hierarchy in the upper left corner of the view. A box representing the current position of the window can be dragged by holding down the left mouse button while moving the mouse. This item does not require a current selection.

7 Known Bugs

7.1 Performance

QDES is not a production tool. It is slow. It was designed as a temporary, somewhat better than straight text, mechanism for creating simple test cases for PDES/STEP tools; hence the name Quick-and-Dirty. Speed was not a concern at any point during the editor's development; given that this is a very large, interpreted system, abysmal performance is to be expected.

7.2 Notes That Missed the Boat

- Smalltalk-80 has the unfortunate habit of displaying the underscore character (_) as a left-arrow (←). Don't be surprised when various names throughout the system look a bit odd because of this.
- In order to avoid various name conflicts, the Smalltalk-80 class corresponding to a particular Express class has `Pdes` prepended to its name, and all underscore characters removed. The appearance of one form or the other throughout the system is fairly unpredictable.

7.3 Things That Are Just Plain Wrong

- Collapsed Browsers do not completely ignore updates: the selection pane of a collapsed Browser is still redisplayed after an update, even though it should not be visible. This kind of damage can be repaired by picking "restore display" from the system menu.
- Overlapping windows are not handled correctly during updates: background windows may not be clipped by foreground windows. Again, this damage is repaired by a "restore display".

- Sometimes QDES gets into an odd state, in which newly created entities do not show up in Browsers. Another symptom of this state of affairs is that deleted entities do not disappear from any Browser. If picking "update browser" does not do away with the problem, click on the "Bug Workaround" icon. Select the text in this workspace and pick "do it" from the middle-button menu. You may have to update each Browser again, but this should fix the problem. Be careful not to close the "Bug Workaround" window - collapse it back to an icon instead.

A References

[Altemueller88] Altemueller, J., The STEP File Structure, ISO TC184/SC4/WG1 Document N279, September, 1988

[Clark90a] Clark, S. N., An Introduction to The NIST PDES Toolkit, NISTIR 4336, National Institute of Standards and Technology, Gaithersburg, MD, May 1990

[Clark90b] Clark, S.N., Fed-X: The NIST Express Translator, NISTIR, National Institute of Standards and Technology, Gaithersburg, MD, forthcoming

[Clark90c] Clark, S.N., The NIST Working Form for STEP, NISTIR 4351, National Institute of Standards and Technology, Gaithersburg, MD, June 1990

[Clark90d] Clark, S.N., QDES Administrative Guide, NISTIR 4334, National Institute of Standards and Technology, Gaithersburg, MD, May 1990

[Mitchell90] Mitchell, M., Y. Yang, S. Ryan, and B. Martin, Data Model Development and Validation for Product Data Exchange, NISTIR 90-4241, National Institute of Standards and Technology, January 1990

[Schenck89] Schenck, D., ed., Information Modeling Language Express: Language Reference Manual, ISO TC184/SC4/WG1 Document N362, May 1989

[Smith88] Smith, B., and G. Rinaudot, eds., Product Data Exchange Specification First Working Draft, NISTIR 88-4004, National Institute of Standards and Technology, Gaithersburg, MD, December 1988

B A QDES Tutorial

Following along with this tutorial in QDES should clear up most fuzzy areas, particularly with respect to idiosyncracies of the Smalltalk-80 user interface. Readers often find that the tutorial is much more useful if it is actually carried out in QDES as it is read. Please note that some of the operations have not been performed in the most efficient way, but rather are somewhat contrived, in order to show the features of QDES.

First start up the QDES demo image:

```
% qdes-demo
```

This starts QDES using the Tokyo IPIM, on which this tutorial relies. It also copies the sample product model used below into the file `tutorial.st` in the current working directory.

QDES should start up with a System Browser visible. If there is no Browser, create one by selecting "browse all" from the system menu. This menu is selected by clicking down on the middle mouse button when the mouse is not in any window. Drag the mouse pointer to the "browse all" option and release the button. The mouse cursor changes to an upper-left-corner shape. Press and hold any mouse button to position the Browser window. The mouse cursor now becomes a lower-left-corner shape; drag this corner to size the window, releasing the mouse button to complete the creation of the Browser.

If the System Browser shows any entities in the selection pane, delete them by picking "clear workspace" from the system menu. This allows you to begin the tutorial with a clean slate.

Next, you will load an existing model which defines some geometry and topology, and add a simple tolerance to the model. Before loading the model, however, you should disable the automatic display update function. To do this, select "display update->disable" from the system menu.

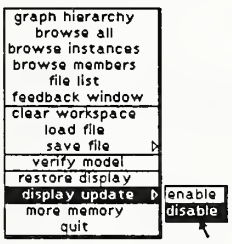


Figure 15: Disabling Automatic Display Update

To confirm your action, you might want to pick "display update" from the system menu (in this case, do not select an item from the sub-menu). The message "display update is disabled." should appear in the Feedback window.

Having disabled automatic display update, go ahead and load the PDES model: select "load file" from the system menu. When QDES asks for the name of the file to load, type "tutorial.st" and hit <return>. Note that this file was originally produced from a STEP file by running:

```
stepparse_qdes -e ipim.exp -s tutorial.step
```

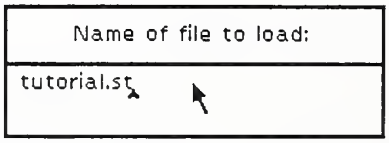


Figure 16: Typing a File Name into a Prompter

The Feedback window immediately informs you that QDES is "loading from file tutorial.st ..."; after grinding away for a while, the Feedback window will say "done in <n> seconds." At this point, the file has been successfully loaded.

Next, pick "display update->enable" from the system menu. This will reconnect the Browser to QDES' internal data structures, and (after a processing delay) update the Browser to show all of the entities loaded.

To get used to the basic feel of Smalltalk and of QDES, you might try scrolling around in the selection pane and picking different entities (with the left mouse button) to see their definitions. Don't worry about corrupting the model: this Browser is in read-only mode (unless you've changed that, in which case you're on your own!).

In poking around the Browser, you have probably noticed that the structure of the model is not crystal-clear. In fact, it's not even obvious what the "top-level" entity is (i.e., the entity which ultimately contains all of the other entities). In this particular case, the top-level entity is a shell. In fact, this will usually be the case when a geometric and topological model is loaded. To find this shell, let's use a Hierarchy Graph View.

Above the Feedback window is a small box (icon) labelled "Topology Hierarchy." Double-click on this icon, and it will expand into a Hierarchy Graph of the topology section of the IPIM. This view has been created especially for this demo image, and will not generally be present in other QDES images. It could be created by spawning a hierarchy graph, finding and selecting the box for "PdesTopology," and spawning a subgraph from this.

Since we know that the entity we want is a shell, we will spawn a Browser containing all of the shells in the model. To do this, first pick the box labelled "shell" in the Graph. The box will be highlighted, indicating that this is the entity class on which we are operating. Now, pick "members" from the middle-button menu in this window, and position and size the new Browser window as before.

We must ask for all members of the abstract class `shell`, because we don't know what specific type of shell we are looking for. This option gives us all instances of `shell` and of any class descended from it. Since the example solid has no interior pockets (it is, in fact, a cube with a cylindrical hole drilled through it), it consists of a single shell, which appears as the sole entity in the selection pane of the new (Filtered) Browser.

Had we not known enough about the model to deduce that the top-level entity was a `shell`, it would have been considerably more difficult to find this entity (although one might argue that the need to find this entity is in fact an artifact of a contrived example ...). One approach would be to pick some entity at random; "browse references" to spawn a Browser on entities which contain it; and then repeat this procedure, selecting some entity at random from each newly spawned Browser, until an entity is found with is referred to by nobody.

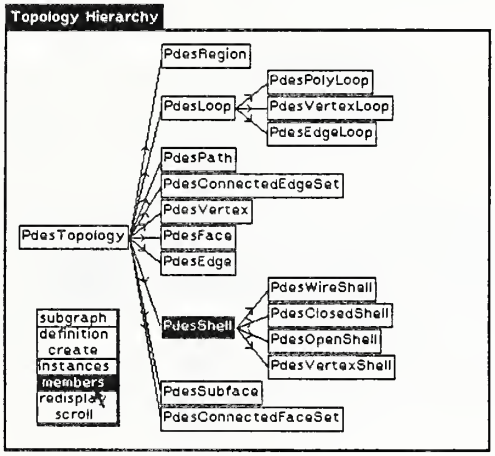


Figure 17: Browsing All Shells

Pick the single shell in the selection pane. Its definition appears in the form pane, and consists of a single attribute, `cshellBoundary`, which is a set of `face_logical_structures`. As an exercise in using QDES to traverse a model, let's find a loop (starting from this set of `face_logical_structures`) and create an Entity Graph View on it.

To do this, first spawn a (Filtered) Browser on the set of faces in the shell: pick "elements" from the middle-button menu in the attribute label pane for `cshellBoundary`. Again, you must specify the position and size of the new Browser with the mouse.

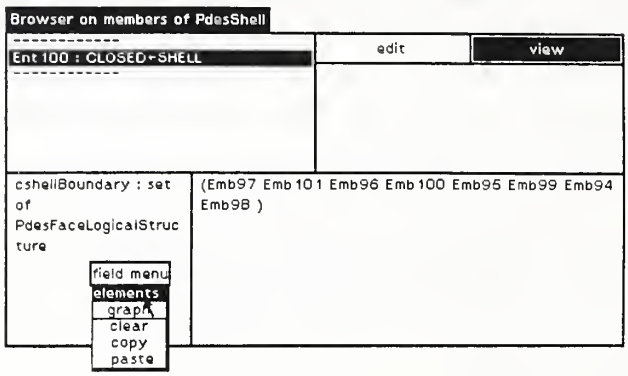


Figure 18: Browsing an Aggregate Field

In this new Browser, pick one of the `face_logical_structures` (in the text, we use `Emb101`) to browse. Its definition appears, and includes a field called `faceElement`. Spawn an Entity Browser on the face (`Ent99`) by selecting "view" from the menu in the attribute label pane.

`Ent99` has an attribute `bounds`, which is a set of `loop_logical_structures`. Browse this set (pick "elements" from the attribute label menu). This last Browser should list two entities (`Emb92` and `Emb91`) in the selection pane, indicating that the face we have been examining is bounded by two edge loops. It happens that `Emb92` consists of the shorter of these loops; it's probably better to pick it rather than `Emb91` (`Emb91` produces a graph layout which does not conveniently fit on the screen).

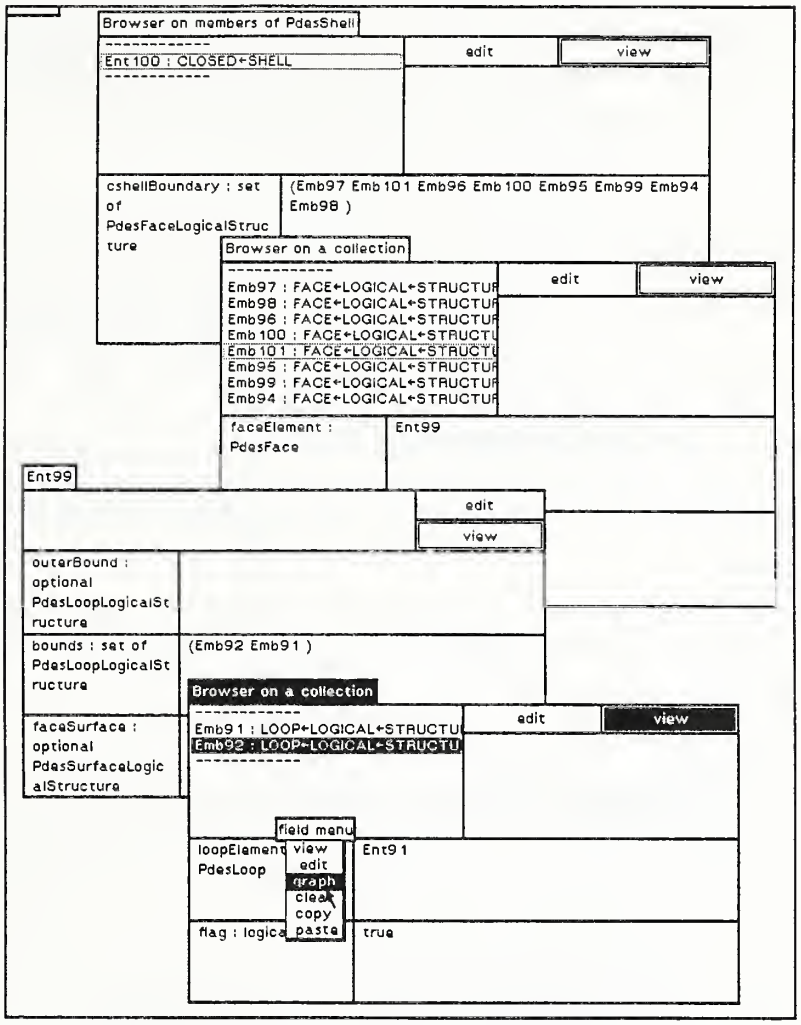


Figure 19: Traversing From A Shell To A Loop

When you pick `Emb92`, its definition appears in the form pane. This definition consists of a logical value (sense) and a loop, which was our ultimate goal. To create an Entity Graph View of this loop, go to the attribute label and pick "graph" from the midle-button menu. When you are asked to size the new window, make it very large: even for `Emb92`, the resulting graph is too wide to fit on a single screen. To see an overview of the full graph, pick "scroll" from the middle-button menu in the Graph View. In the upper left corner of the window, a small image of the graph appears, along with a box indicating the current portion which is visible in the window. If you click and hold the left mouse button, you can drag this box around. When you release the button, the newly selected portion of the Graph will be displayed.

As you may be able to see from the Graph, this particular loop (`Ent91`) consists of only two edges; we see that it must be the loop around one end of the cylinder (the modeler used to create this model represents cylinders as being split parallel to the axis into two separate surfaces, resulting in two semi-circular edges at each end).

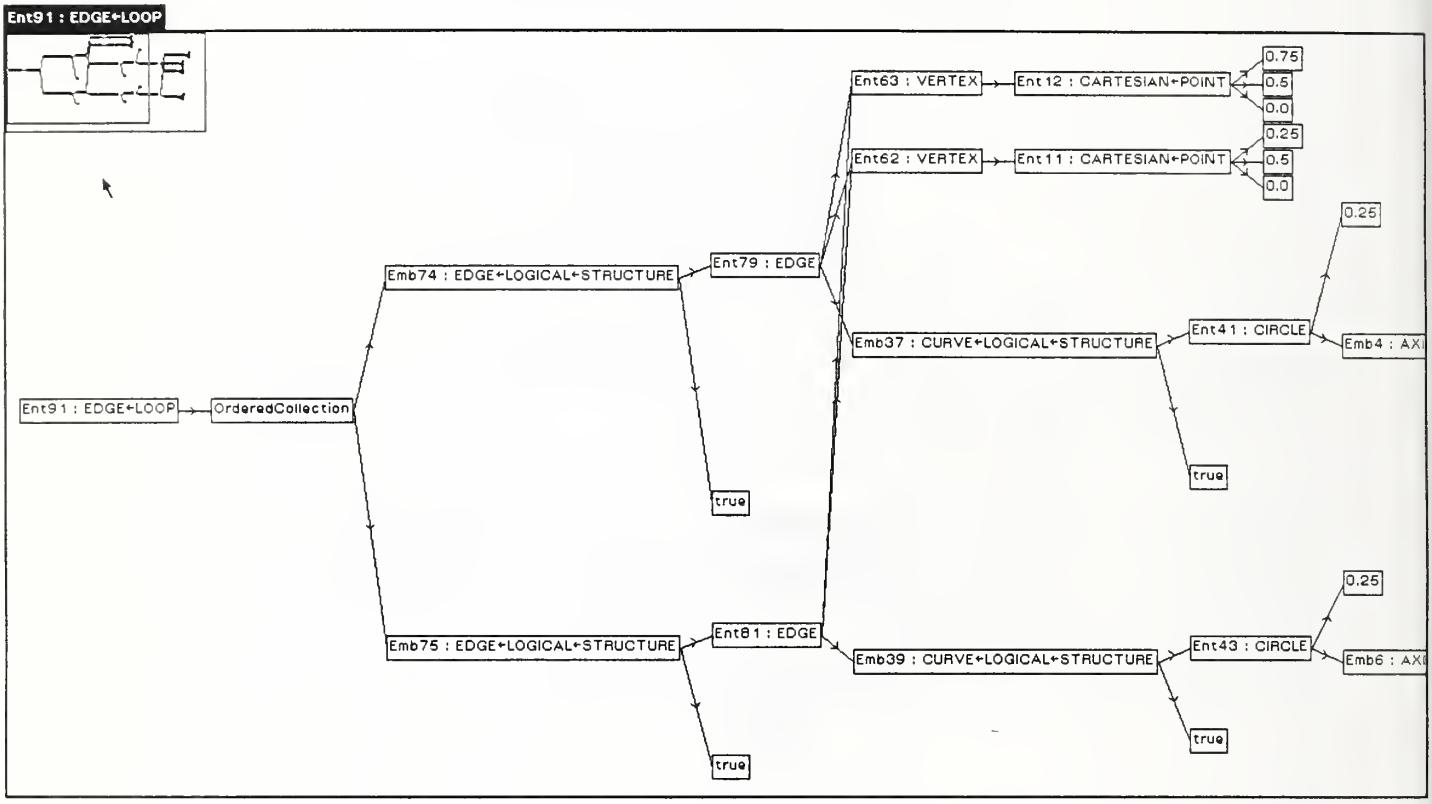


Figure 20: Graphical View of an Edge Loop

For the second part of the tutorial, you will create a cylindricity tolerance on the inside of the hole in the block. In order to do this, you must change the System Browser to edit mode by clicking in the box marked "edit" in the upper right corner. You should probably also close all of the Browsers (except the System Browser) created in the previous exercise, and collapse the Topology Hierarchy Graph back to its icon (both "close" and "collapse" are items on the right-button menu in every window).

First, create a blank cylindricity tolerance: in the selection pane of the system, pick "create entity." A large menu appears, listing all of the PDES Entity classes which have no superclasses. Holding down the left mouse button, search this (alphabetical) menu for `PdesTolerance`, which is probably not immediately visible. To scroll the menu upwards (in order to find `PdesTolerance`), move the mouse down past the bottom of the menu while still holding the left button down. When you find `PdesTolerance`, notice that it is marked with an open triangle in the right margin. This indicates that there are subclasses of `PdesTolerance`. Move the mouse over

this triangle to the right, and a menu of these subclasses (of which there is, in fact, only one, `PdesShapeTolerance`) appears. Moving to the right again gives a menu with its single subclass `PdesGeometricTolerance`. Moving over the triangle on this menu at last gives us a menu which includes `PdesCylindricity`, which you should choose (by releasing the mouse button). QDES then asks for the name of the new entity; type "tolerance" and hit <return>.

When the new tolerance has been created, its definition appears in the form pane of the System Browser. We see that there are two attributes to fill in: a set of toleranced entities, and a magnitude. Let's do the easy one first: the magnitude. Pop up the middle-button menu in the label pane for this attribute, and select "edit." Since the attribute currently has no value, QDES creates a new `tolerance_magnitude` and asks for its name. Call it "magnitude." You must then specify a window for an Entity Browser on this new entity, which, as it turns out, has only a single attribute, a real number. Since this Browser was spawned by an editable Browser, it is also editable. You can now put the mouse in the value pane for the tolerance magnitude, type in a real number, and hit <return>. The magnitude is now specified. You can close the Entity Browser on `magnitude`.

The set of toleranced entities is considerably more complex to specify. The elements of this set are of type `area_or_fos`, which is a select type (consisting of `area_shape_element` and `feature_of_size`). Since QDES cannot automatically create instances of a select type, we must first create any values to be put into the set, and then actually put them in. For this particular tolerance, we need a single `area_shape_element` which includes both faces which make up the surface of the hole. Let's create this object: as before, pick "create entity" from the selection pane menu, and pick `PdesAreaShapeElement` from the menu which appears (hint: it is a subclass of `PdesDimensionality2ShapeElement`, which is a subclass of `PdesShapeElement`). Call the new entity "areaShape."

Next, scroll to the bottom of the selection pane and select `tolerance` again. Go to the value pane for the attribute `toleranceEnts`, type in "(areaShape)", and hit <return>. Note the parentheses in the value: without them, the value typed would simply be an entity, while what is expected is an aggregate of zero or more entities.

We return to the definition of `areaShape` by a somewhat circuitous route. The method shown is not terribly appropriate to the problem at hand, but demonstrates a technique which may be quite useful in other circumstances. First, double-click on the word `areaShape`. The entire word should be highlighted; if not, try again (remember that Smalltalk only considers the physical distance between clicks; it doesn't care how long you wait in between). Next, pick "copy" from the middle button menu in this value pane. "areaShape" is now on the clipboard.

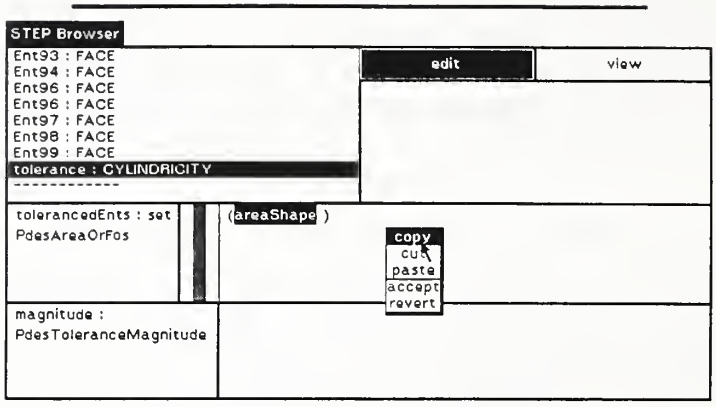


Figure 21: Copying a Text Selection

Now for the magic: from the selection pane menu, pick "find entity." A Prompter appears, asking for the name of the entity to find. The middle button menu in the Prompter is similar to that in an attribute value pane; in particular, you can pick "paste," and the word you just copied ("areaShape") is inserted into the Prompter. Hit <return>, and that entity is selected. Although the copy-and-paste rigamarole was clearly a waste of time in this case, it demonstrates the flexibility of the copy-and-paste mechanism in QDES.

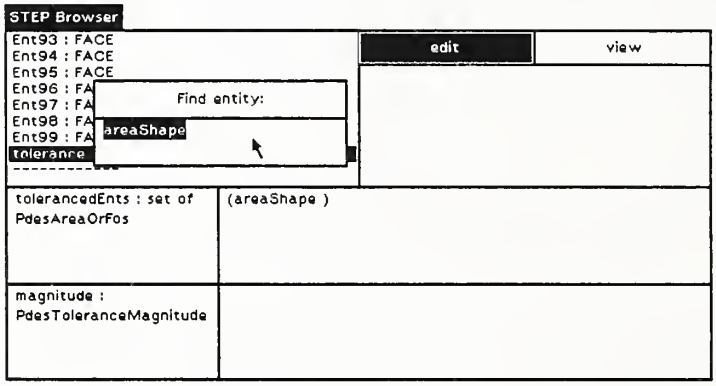


Figure 22: Pasting Into the Find Entity Prompter

Examining the form for `areaShape`, we find room for an optional form feature (which is not needed in this case) and for a list of `area_representations`, which we must provide. For starters, just make a list of one `area_representation`, say, "areaRep." To do this, type "(areaRep)" into the value pane for this attribute. When you hit <return>, QDES will create an empty `area_representation` called "areaRep", and add it to the list in the selection pane.

Select this new entity. Now, QDES has created the most general type of object which can fill the `area_representation` role. For our example, we actually need something more specific: a `surface_area_rep`. So we need to change `areaRep` into this more specific type of object. To do this, pick "subclass entity" from the selection pane. A menu pops up with all of the subclasses of `area_representation`. Find `PdesSurfaceAreaRep` and select it. QDES has now changed `areaRep` into a new type of object. Note that all references to this object are still valid, however, since its new type is a subtype of the old.

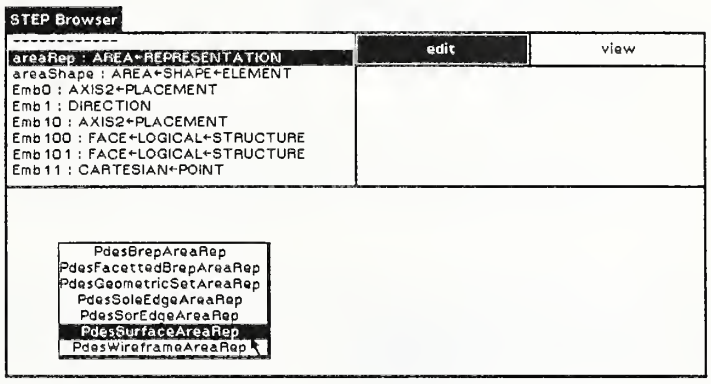


Figure 23: Subclassing an Entity

Look at the form for this new entity. It requires a single `face`, which, in our example, should be one of the cylindrical faces in the hole. We will find these two faces by traversing backwards from the cylindrical surfaces to the faces which contain them. To begin this process, expand the "Geometry Hierarchy" icon above the feedback window. Select "PdesSurface" and spawn a Browser on all members of this class. (Once again, note that this view will not, in general, be present in a standard QDES image).

In the selection pane of the new Browser, you will see several `planes` and two `cylindrical_surfaces` (the latter being `Ent49` and `Ent50`). Select the first of these and pick "browse references" from the selection pane menu. QDES spawns a new Browser which lists the single entity which refers to `Ent49`, a `surface_logical_structure` called `Emb88`. Repeat the process to get the references to this entity; the resulting Browser should list one `face`, called `Ent97`. This is the face we want. (You will eventually want to repeat this process to find the `face` that refers to `Ent50`, which turns out to be `Ent98`).

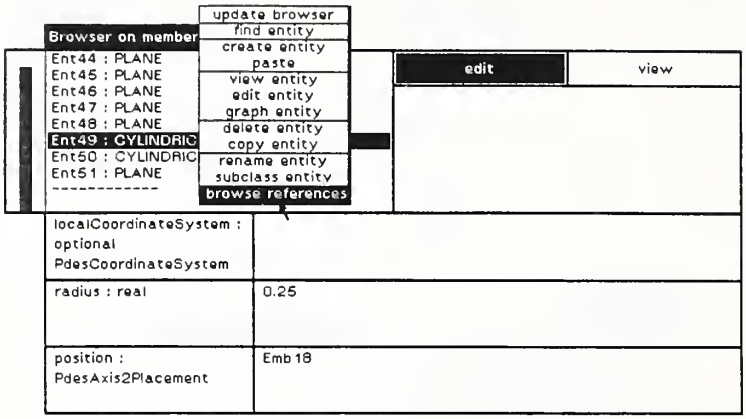


Figure 24: Browsing All References to an Entity

Select `Ent97` in this last Browser and choose "copy entity" in the selection pane. The object which is `Ent97` is now on the clipboard). Now, go back to the System Browser, where the current entity should be `areaRep`. From the menu in the label pane for the `defintion` attribute, pick "paste," which pastes the entity on the clipboard into that attribute. (As an aside, note that "defintion," clearly a typo, is actually the name of this attribute in the Tokyo IPIM, demonstrating the direct connection between the Express specification and the data structures maintained by QDES).

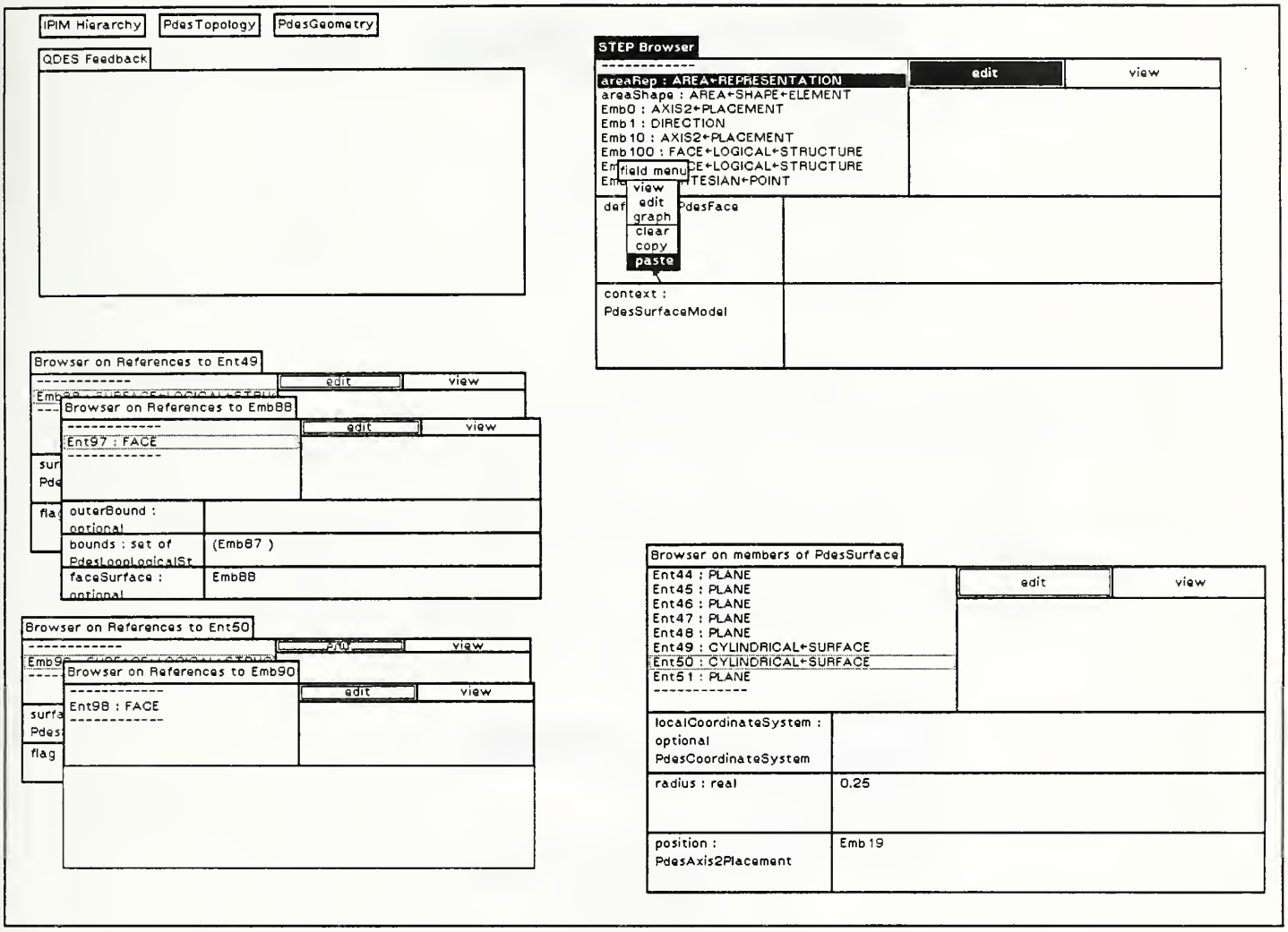


Figure 25: Pasting an Entity Into an Attribute

Looking at the definition of `areaRep`, we realize that we need two `area_representations` in order to include both cylindrical faces in the tolerance. Let's plan to call them `areaRep97` and `areaRep98`. Thus, we first need to rename `areaRep` to be `areaRep97`. To do this, make sure that `areaRep` is selected in the System Browser, and pick "rename entity" from the selection pane menu. Type "areaRep97" into the Prompter which appears, and the deed is done.

Now, we need another `surface_area_rep` which is very similar to this one. So let's make a copy: pick "copy entity" from the selection pane menu. Now, pick "paste" from this same menu. QDES prompts for a name, which should be "areaRep98," and creates the new entity. Note that pasting into the selection pane is different from pasting into an attribute: in the latter case, we are only pasting a reference to whatever entity is on the clipboard, while in the former case, we are actually creating a whole new entity.

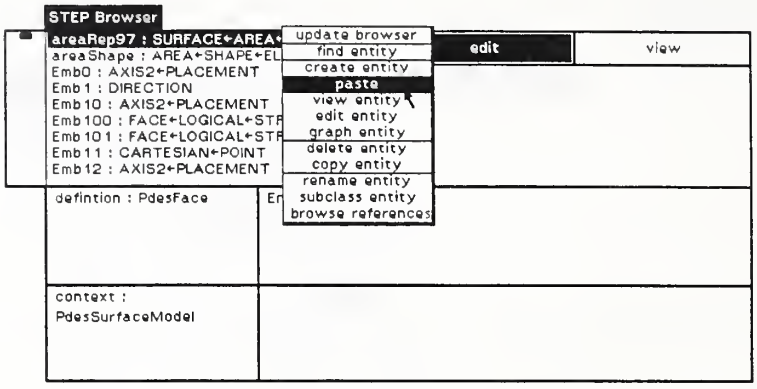


Figure 26: Making a Copy of an Entity

This new entity actually needs to be slightly different, as it refers to a different face. To make this change, go the attribute value pane for the `defintion` attribute of `areaRep98`, change "Ent97" to "Ent98", and hit <return>. (Just for kicks, try hitting return when the value reads "Ent9." You should get an error Notifier: `Ent9`, being a `cartesian_point`, cannot be put into this attribute. Click any button to proceed).

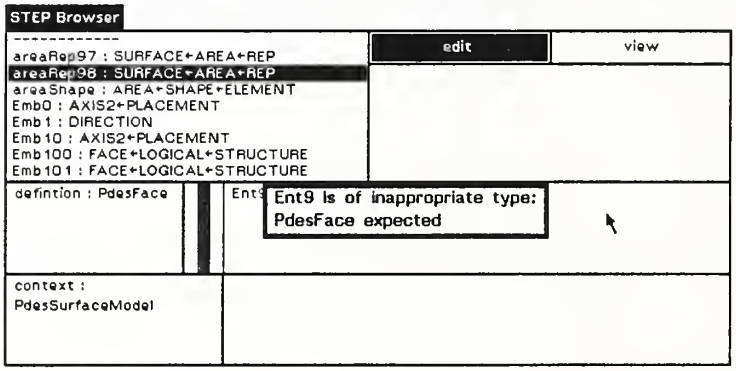


Figure 27: A Type Mismatch Error Notifier

We are almost done. All that remains is to add `areaRep98` to the list of toleranced entities in `areaShape`. Select `areaShape` in the selection pane of the System Browser. Note that the change in the name of `areaRep`[97] is reflected in the form which appears. Add the new entity name to the list, which should now read "(areaRep97 areaRep98)," and hit <return>. You might create an Entity Graph View of the complete `tolerance` entity, but it will be quite large. The image below was created by cheating some in Smalltalk.

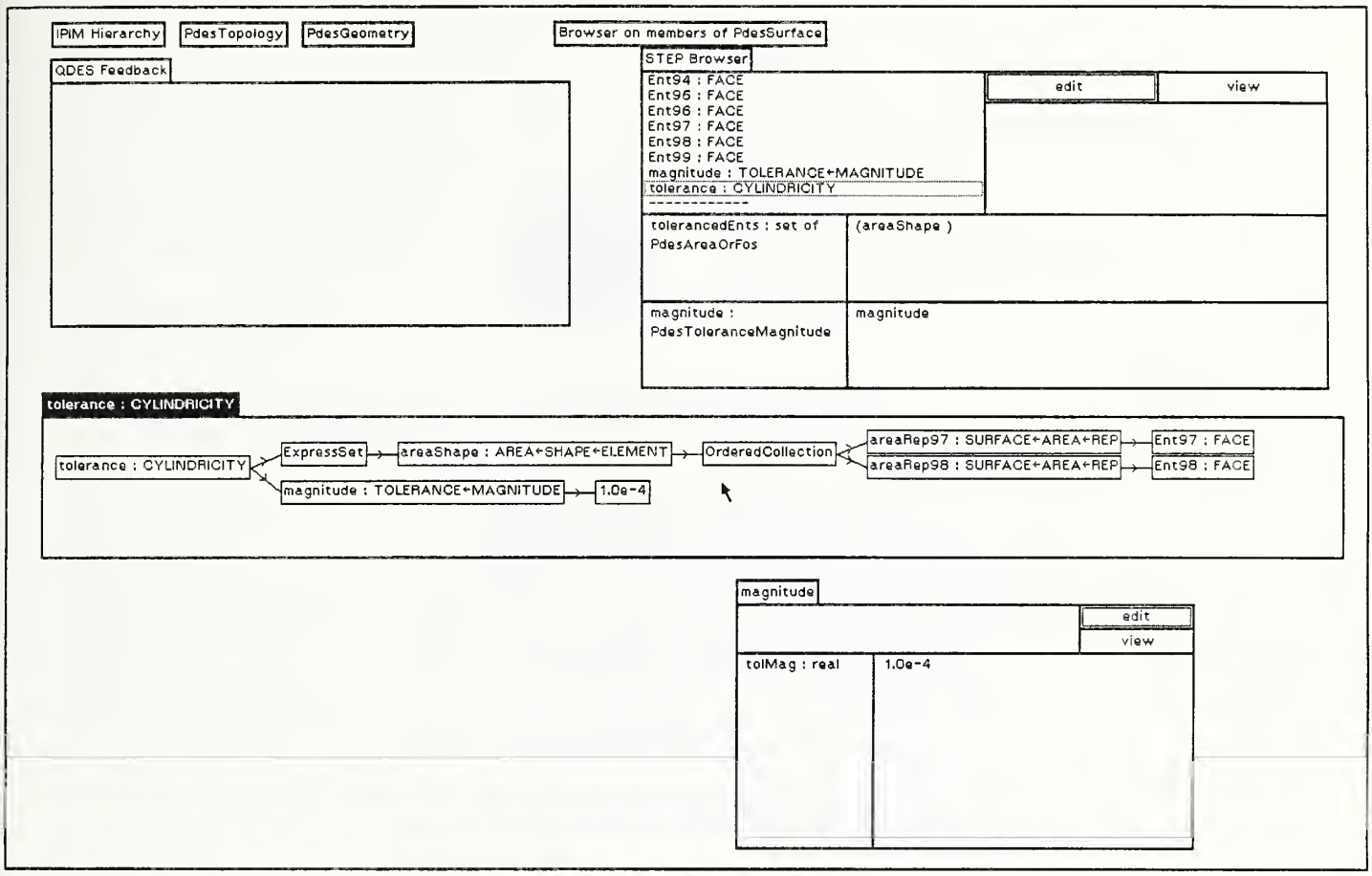


Figure 28: The Final Result

ORDER and INFORMATION FORM

MAIL TO:

National Institute of Standards and Technology
Gaithersburg MD., 20899
Metrology Building, Rm-A127
Attn: Secretary National PDES Testbed
(301) 975-3508

Please send the following documents and/or software:

- ☐ Clark, S.N., An Introduction to The NIST PDES Toolkit, NISTIR, National Institute of Standards and Technology, Gaithersburg, MD.
- ☐ Clark, S.N., The NIST PDES Toolkit: Technical Fundamentals, NISTIR, National Institute of Standards and Technology, Gaithersburg, MD.
- ☐ Clark, S.N., Fed-X: The NIST Express Translator, NISTIR, National Institute of Standards and Technology, Gaithersburg, MD.
- ☐ Clark, S.N., The NIST Working Form for STEP, NISTIR, National Institute of Standards and Technology, Gaithersburg, MD.
- ☐ Clark, S.N., NIST Express Working Form Programmer's Reference, NISTIR, National Institute of Standards and Technology, Gaithersburg, MD.
- ☐ Clark, S.N., NIST STEP Working Form Programmer's Reference, NISTIR, National Institute of Standards and Technology, Gaithersburg, MD.
- ☐ Clark, S.N., QDES User's Guide, NISTIR, National Institute of Standards and Technology, Gaithersburg, MD.
- ☐ Clark, S.N., QDES Administrative Guide, NISTIR, National Institute of Standards and Technology, Gaithersburg, MD.
- ☐ Morris, K.C., Translating Express to SQL: A User's Guide, NISTIR, National Institute of Standards and Technology, Gaithersburg, MD.
- ☐ Nickerson, D., The NIST Database Loader: STEP Working Form to SQL, NISTIR, National Institute of Standards and Technology, Gaithersburg, MD.
- ☐ OTHER (PLEASE SPECIFY)

_____

_____

Some documents and software will be available from NTIS. When available, that ordering information will be forwarded.

NIST

NIST-114A
(REV. 3-89)

U.S. DEPARTMENT OF COMMERCE
NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY

BIBLIOGRAPHIC DATA SHEET

1. PUBLICATION OR REPORT NUMBER
NISTIR 4361

2. PERFORMING ORGANIZATION REPORT NUMBER

3. PUBLICATION DATE
JULY 1990

4. TITLE AND SUBTITLE

"QDES User's Guide"

5. AUTHOR(S)

Stephen Nowland Clark

6. PERFORMING ORGANIZATION (IF JOINT OR OTHER THAN NIST, SEE INSTRUCTIONS)

U.S. DEPARTMENT OF COMMERCE
NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY
GAITHERSBURG, MD 20899

7. CONTRACT/GRANT NUMBER

8. TYPE OF REPORT AND PERIOD COVERED

9. SPONSORING ORGANIZATION NAME AND COMPLETE ADDRESS (STREET, CITY, STATE, ZIP)

10. SUPPLEMENTARY NOTES

[ ] DOCUMENT DESCRIBES A COMPUTER PROGRAM; SF-185, FIPS SOFTWARE SUMMARY, IS ATTACHED.

11. ABSTRACT (A 200-WORD OR LESS FACTUAL SUMMARY OF MOST SIGNIFICANT INFORMATION. IF DOCUMENT INCLUDES A SIGNIFICANT BIBLIOGRAPHY OR LITERATURE SURVEY, MENTION IT HERE.)

The Product Data Exchange Specification (PDES) is an emerging standard for the exchange of product information among various manufacturing applications. The neutral exchange medium for PDES product models is the STEP physical file format. The National PDES Testbed at NIST has developed QDES, a window-based editor for STEP product models. The editor, written in Smalltalk-80, is schema-driven; in the Testbed context, an Express information model is used to describe the objects to be manipulated; QDES itself thus has no a priori knowledge of its domain. This document describes the operation of the editor. A tutorial is included.

12. KEY WORDS (6 TO 12 ENTRIES; ALPHABETICAL ORDER; CAPITALIZE ONLY PROPER NAMES; AND SEPARATE KEY WORDS BY SEMICOLONS)

data modeling; Express; PDES; QDES; schema-independent software; Smalltalk; STEP

13. AVAILABILITY

[X] UNLIMITED

[ ] FOR OFFICIAL DISTRIBUTION. DO NOT RELEASE TO NATIONAL TECHNICAL INFORMATION SERVICE (NTIS).

[ ] ORDER FROM SUPERINTENDENT OF DOCUMENTS, U.S. GOVERNMENT PRINTING OFFICE, WASHINGTON, DC 20402.

[X] ORDER FROM NATIONAL TECHNICAL INFORMATION SERVICE (NTIS), SPRINGFIELD, VA 22161.

14. NUMBER OF PRINTED PAGES
41

15. PRICE
A03

ELECTRONIC FORM