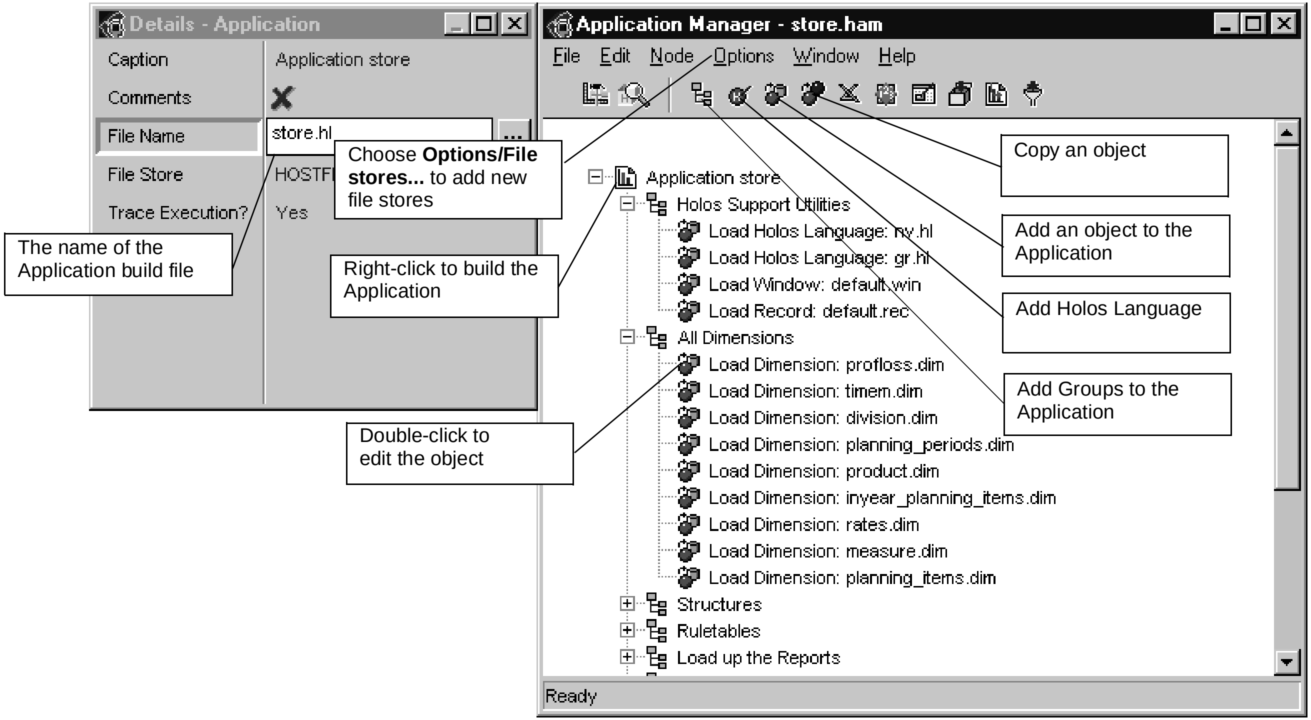
The Application Manager



Application store
- Holos Support Utilities
  - Load Holos Language: nv.hl
  - Load Holos Language: gr.hl
  - Load Window: default.win
  - Load Record: default.rec
- All Dimensions
  - Load Dimension: profloss.dim
  - Load Dimension: timem.dim
  - Load Dimension: division.dim
  - Load Dimension: planning_periods.dim
  - Load Dimension: product.dim
  - Load Dimension: inyear_planning_items.dim
  - Load Dimension: rates.dim
  - Load Dimension: measure.dim
  - Load Dimension: planning_items.dim
- Structures
- Ruletables
- Load up the Reports

Choose **Options/File stores...** to add new file stores

The name of the Application build file

Right-click to build the Application

Double-click to edit the object

Copy an object

Add an object to the Application

Add Holos Language

Add Groups to the Application

The Application Manager (cont'd)

You can copy and paste objects between Application Manager sessions

You can copy and paste objects between Application Manager sessions using edit/copy in one session, opening a new .ham file and then using edit/paste. If the objects do not appear, you probably have no object selected in the new session. Select the object below which you want the new objects to appear then try again.

Common properties of objects can be changed together

For example, if you want to change the file store for a group of objects, selecting all the relevant objects (using the standard multiple selection methods), followed by the Details sheet, will show all the properties common to those objects, including file store, and one change will affect them all.

Always use the default file extensions unless there is a good reason not to

By default, the Application Manager uses the file extensions documented under **Basics file extensions** in the server help. Although you **can** use alternative extensions, you should always use those documented, unless there is a pressing reason not to, as it makes using the Object Viewer much easier.

Use file stores to develop all applications

When using the Application Manager, it is best always to use a file store to store your objects. Whilst it is possible to use the current host directory (using HOSTFILE) this may cause problems if the source control (checking in/checking out) system is used.

Avoid using file store names starting with 'holos_' except for those beginning 'holos_share'

Avoid using file store names which start with 'holos_' as the Application Manager treats these as read-only (to protect holos_support, holos_examples etc).

Obviously the only exception to this is when using file stores starting with 'holos_share' which invokes the use of the source control systems.

Always add any automatically-loaded utilities to your application tree

If the Report Designer or Worksheet have been called from the Application Manager to create reports that will be referenced in an Application Manager file, remember that Interactive Service utilities may well have been automatically loaded by the Report Designer or Worksheet. These utilities will need to be added within an application file to enable the application to be successfully loaded in a subsequent new Holos session. The files required will vary depending on which Interactive Services are used:

Interactive Service	Utility
Graphics	holos_support:gr.hl
Key Performance Indicators	holos_support:kpi.hl
Regression Modelling	holos_support:mod_reg.hl
Multi Linear Regression Modelling	holos_support:mod_mlr.hl
Modelling	holos_support:mod_out.hl
Forecast Modelling	holos_support:mod_forc.hl

Interactive Service	Utility
Box Jenkins Modelling	holos_support:mod_box.hl
All Modelling Types	holos_support:mod_all.hl
Navigation	holos_support:nv.hl
SQL Record Drill navigation	holos_support:nv_sql.hl

The PCFILE ‘template.ham’ will be used as an initial template for new applications

This is useful if you always want to load a set of standard tools into you applications.

The easiest way to do this is to create the objects you require then save the tree (to any file store) as *template.ham*. As the saving process leaves a copy of this file on the PC you can now delete the version copied to the server.

Always use ‘Test if Loaded’ on holos_support objects (and make sure you use a valid name)

As many of the *holos_support* utilities load each other, always ensure the 'Test if Loaded?' property sheet item is set for them. However, for this to work you need to make sure that the ‘name’ property gives the name of one of the objects in the file and not the file name (which it is the default).

For example if you add an object for ‘holos_support:set.hl’, the default name property will be set and the Application Manager will perform the following test

```
IF NOT DEFINED(set)
   EXECUTE "holos_support:set.hl"
END IF
```

However ‘set.hl’ has no object in it called ‘set’, therefore this will not work properly. However, ‘set.hl’ does have an object called set$_unload in it so if the name property is changed to this then everything should be okay.

How do I choose between HL files and HL objects?

Small fragments of HL code can be included directly in a Holos Language object (for example logging onto a database) but more significant amounts of code can be controlled and maintained more easily as an HL hostfile included into the application in an object. (Also remember that code in an HL object is included in the ‘.ham’ file and this is copied to and from the server when it is opened and closed).

How do I access dynamically-created objects from within the Application Manager?

A dynamically created object (such as a snapshot report in the worksheet) can be added to an Application Manager tree by adding a new node for the object, filling out the properties as usual, but making sure that the name you type matches that of the dynamically created object. If the editor mode is now set to ‘Memory edit’, the Application Manager will edit the memory copy of the object.

Notes

- The object will be saved to the file you have specified when you close the edit session.
- If you leave the editor mode as file edit then the memory object will get overwritten before it is read into the editor.

The Application Manager (cont’d)

How do I use an existing source control system?

It is possible to use an external source control system (for example SCCS, common on UNIX) in place of the Application Manager routines. This is done by writing functions that simulate the following Application Manager functions:

```
am$$scstest
am$$scsget
am$$scsput
am$$scscancel
```

These functions should be registered as call-backs in the following variables:

```
am$_scstest_fnc
am$_scsget_fnc
am$_scsput_fnc
am$_scscancel_fnc
```

Holos will then call these functions rather than its own routines. For more details, see the server help under ‘utilities am.hl SCCS’.

How do I stop trace messages being written by an Application Manager build file?

When an Application Manager build file runs, it writes out a series of debug statements of the form

```
DEBUG: 1
DEBUG: 2
DEBUG: -1
DEBUG: 3
```

This is to allow the Application Manager to show the execution of the file as it runs (by putting the green ticks on each of the executed nodes). You can stop this happening by setting the 'Trace Execution?' property of the application (on the top node) to ‘No’.

How do I use the source control system?

Define file stores (from the Options File Stores dialog) with the prefix ‘holos_share’. This will signal to the Application Manager that the source control system is to be used. The directory or directories pointed to by these file stores will be treated as a shared source code library by the Application Manager, which will control the checking in and checking out of the files.

When using the source control system it is important to make sure that developers each have their own default directory (as this is where the checked-out files are stored).