

2.1 Overview

The User Interface facilities cover all aspects of the user interface. They include the enablers used to deliver a user interface and the tools used by application developers to develop that interface. It also includes the facilities needed to give users easy access to their applications and to automate parts of their work.

Figure 2-1 on page 2-2 illustrates the components of a user interface. The components are as follows:

The user interface style defines the look and feel of the user interface in terms of user interface objects (for example, a menu bar) and the actions users can perform on these objects. It also prescribes the use of the workstation hardware.

The workstation hardware includes the equipment used to deliver the user interface, that is screen, keyboard, mouse, printer and security devices. It is described in terms of its functional capability and also its physical properties (for example, screen luminance and keyboard layout).

User interface enablers deliver the user interface to a range of applications. There are several types of enabler, including window managers, terminal emulator programs, and class libraries of user interface objects.

User interface enablers are grouped into three main Common Facilities:

- Rendering management facility, which provides access to and abstractions of the user interface hardware, support for windows, user interface objects and dialogue objects.

- Compound presentation facility, which provides a framework for subdividing a display window into multiple parts and maps the display portion of a compound document into them.

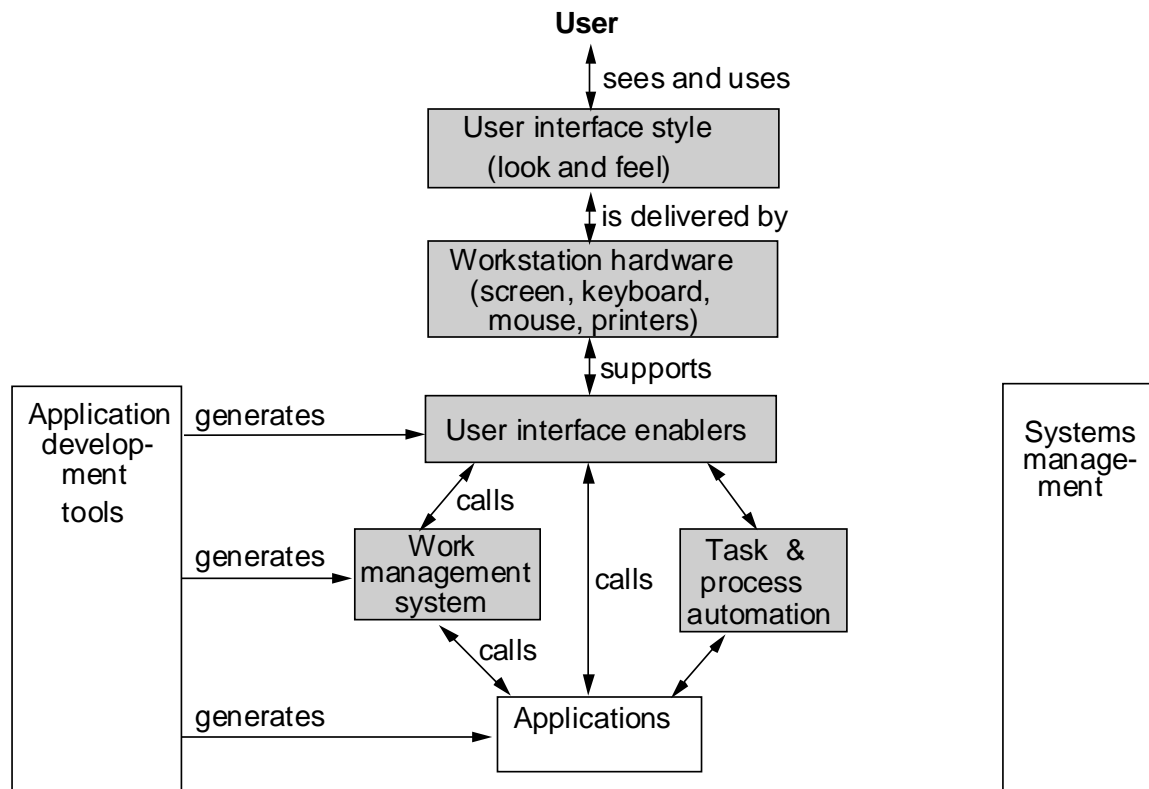


Figure 2-1 Components of the User Interface Facility

- User support facilities, which provide facilities that are common across applications such as help and text checking.

The work management system enables an information system to maintain the user's working context. The capabilities of this component include:

- Support for user single system log-on.
- Definition of the user's working environment in terms of the applications and information used daily.
- Desktop management which provides iconic desktops as a means of visualising the user's working environment.

Task and process automation provides facilities to allow users to:

- Automate their tasks through the use of scripting.
- Automate desk procedures by writing descriptions of those procedures in a simple language.
- Participate in enterprise business procedures by using workflows.

2.2 *Rendering Management*

2.2.1 *Description and Requirements*

Rendering Management provides facilities to present information for output on devices such as screens, printers, plotters and sound and speech output devices. It also provides facilities to handle user input from a variety of different hardware devices such a keyboard, mouse, scanners, speech recognition devices, and security devices.

Rendering Management includes support for:

- Window management.
- Class libraries for user interface objects.
- User interface dialogue objects (for example, menu bars, scroll bars, and so forth).
- Abstractions of the many different input and output devices.

2.2.2 *Related Standards and References*

Many user interface styles define the user interface objects that need to be supported by rendering management. These styles include:

Apple Computer, *Macintosh Human Interface Guidelines*, Addison-Wesley Publishing Company, Reading, MA 1992.

International Business Machines (IBM), *Systems Application Architecture, Common User Access Guide to User Interface Design*, October 1991.

International Business Machines (IBM), *Systems Application Architecture, Common User Access Advanced Interface Design Reference*, October 1991.

Microsoft Corporation, *The Windows Interface: An Application Style Guide*, Microsoft Press, Redmond WA 1992.

Microsoft Corporation, *The Microsoft Windows User Interface Design Guide* (draft), 1992.

NeXT Computer Inc., *NEXTSTEP User Interface Guidelines*, November 1993.

NeXT Computer Inc., *NEXTSTEP General Reference, Volume 1*, November 1993.

Open Software Foundation, *OSF/Motif Style Guide (Release 1.2)*, Prentice-Hall, Inc., Englewood Cliffs, NJ 1993.

Sun Microsystems, Inc. *OPEN LOOK Graphical User Interface Application Style Guidelines*, Addison-Wesley Publishing Company, Reading, MA 1990.

Sun Microsystems, Inc. *OPEN LOOK Graphical User Interface Functional Specification Guidelines*, Addison-Wesley Publishing Company, Reading, MA 1989.

2.2.3 Relationship to Other Components of OMA

Rendering management provides a vital link between the hardware and operating system and the Compound Presentation Facility.

2.2.4 Technical Issues

None.

2.3 Compound Presentation Facility

2.3.1 Description and Requirements

The Compound Presentation Facility should provide a framework for sharing and subdividing a display window into multiple parts. These parts may be peers of each other or may in turn may be embedded into other parts. This facility maps to the display portion of a compound document architecture.

There are many issues to be considered in the Compound Presentation facility. The facility should address:

- Geometry management.
- Human interface event distribution.
- Shared human interface control management (for example, menus, palettes, button bars).
- Rendering management (including printing).

2.3.2 Relationship to Components of OMA

None.

2.3.3 *Related Standards and References*

None.

2.4 *User Support Facility*

2.4.1 *Description and Requirements*

Today, a number of facilities and capabilities are provided to the end user by individual applications that are duplicated across applications and provide similar functionality. Although there may be a set of general requirements that are shared among these non-application function specific facilities, they have typically been designed and implemented on a custom basis. This diversity in development results in different presentation schemes, styles and semantics to the user. These facilities also become a static part of the application, increasing the code and data size.

Providing these non-application specific functions as stand-alone, reusable Common Facilities provides advantages to the user and the developer. The user will benefit from consistent interface style, semantics, and predictable behavior. The developer will benefit from reusable code with standardized interfaces.

The number and function of User Support facilities is expected to expand over time as more application logic is abstracted into reusable components. In the future, User Support Facilities will cover the following cross application functions:

- Help
- Text Checking

In the future, User Support facilities will expand to include versioning; annotating; standard text; graph and spreadsheet functions; and other, additional facilities.

Help User Support Facility

The Help system must be stand-alone and provide mechanisms to access, present, and interchange help data, including internationalized text; image; sound; and animation. It must also support multiple presentation styles (such as context-sensitive, a manual browser, and hyper-text/media); a standard storage format (for example, SGML); and multiple storage styles (for example, all help data in a single file, a file per page of display, and so forth).

The Help Facility must include OMG IDL specifications for at least the following interfaces:

- Initializing and freeing the Help Facility object.

- Attaching (and detaching) help objects to any objects in the application so that appropriate help data can be rendered (and nullifying the rendering operation) when the user requests help. This operation may need to take a marker name if the help file contains multiple segments of help data. Time-based data such as sound and animation may need begin-time and end-time information if only a portion of data in the file is to be rendered. An optional parameter may also be needed to indicate the name of other applications that need to be invoked.
- Rendering help objects for programmatically displaying the help data. The parameters are the same as in attaching and detaching.
- Querying for help event history. There should be a top-level help object that manages the sub-components, interaction with other objects, and interface to other OMA components such as event management. If the current help is rendered in response to a user action from another help object (i.e. nested), the information of the level of nesting should be made available to the application on request.
- Querying the current help file path and name.
- Querying the object name to which it is attached.
- Printing to send the currently rendered help data to a hard copy service.

Text Checking User Support Facility

The Text Checking Facility must be stand-alone and provide mechanisms to allow applications to pass strings or files of internationalized text to be subjected to various lexical checks, including, at a minimum, spelling, hyphenation, thesaurus and grammar. The facility must provide the capability to support text strings independent of font and mark-up symbols, and must be able to interact with the application in order to check a single string of text, or an entire file or document.

This facility must include OMG IDL specifications for at least the following interfaces:

- Initializing and freeing the text checking objects.
- Sending and receiving text strings with both the original text and the changed text (to support versioning in the application).
- Provide customization of the service to be executed against the string, including at least: spelling check; hyphenation check; thesaurus; and grammar check.
- Provide capability to query the dictionary in use, and to provide alternative, application-provided dictionaries.

2.4.2 *Related Standards and References*

ISO 9241 *Ergonomic Requirements for Office Work with Visual Display Terminals, Part 13 User Guidance* (not yet published).

ISO/IEC 8879:1986 Standardized Generalized Markup Language (SGML).

IEEE P1201.2 *Recommended Practice for Graphical User Interface Drivability*, Balloting Draft 2, August 1993.

Human Factors Society (HFS) Human-Computer Interaction (HCI) Committee, *User Guidance* (draft), April 1991.

Apple Computer, *Macintosh Human Interface Guidelines*, Addison-Wesley Publishing Company, Reading, MA 1992.

International Business Machines (IBM), *Systems Application Architecture, Common User Access Guide to User Interface Design*, October 1991.

International Business Machines (IBM), *Systems Application Architecture, Common User Access Advanced Interface Design Reference*, October 1991.

Microsoft Corporation, *The Windows Interface: An Application Style Guide*, Microsoft Press, Redmond WA 1992.

Microsoft Corporation, *The Microsoft Windows User Interface Design Guide* (draft), 1992.

NeXT Computer Inc., *NEXTSTEP User Interface Guidelines*, November 1993.

NeXT Computer Inc., *NEXTSTEP General Reference, Volume 1*, November 1993.

Open Software Foundation, *OSF/Motif Style Guide (Release 1.2)*, Prentice-Hall, Inc. Englewood Cliffs, NJ 1993.

Sun Microsystems, Inc. *OPEN LOOK Graphical User Interface Application Style Guidelines*, Addison-Wesley Publishing Company, Reading, MA 1990.

Sun Microsystems, Inc. *OPEN LOOK Graphical User Interface Functional Specification Guidelines*, Addison-Wesley Publishing Company, Reading, MA 1989.

2.4.3 *Relationship to Components of OMA*

The User Support facilities should require no changes to the other OMG standards. The Facilities should use the OMG Object Management Architecture, Object Model, Common Object Request Broker Architecture, Object Services, and other Common Facilities as specified. In particular, the User Support facilities will benefit from other Common Facilities such as User Interface facilities and the Information Management facilities, and may

use and specialize Object Services such as the Event Service. (The Event Service is described in *CORBAservices*.)

2.4.4 Technical Issues

None.

2.5 Desktop Management Facility

2.5.1 Description and Requirements

Desktop Management facilities provide a general purpose infrastructure for visualizing the user's working environment. This infrastructure supports a conceptual model of user objects that is specialized by desktop implementations that support further specialization and instantiation by users, projects, and enterprises.

The fundamental types of user objects are:

Information: objects that are processed by tools in the context of user tasks. An information object can be associated with a workflow, which defines the operations and policies which can be applied to the information. This workflow may be composed of a single operation or many serial and parallel operations sequenced by rules. Information objects may contain application information as well as workflow, model, resource, and project information. Information objects may represent or be part of:

- Aggregations, which group together information in n-level container hierarchies
- Versions, which represent information evolution
- Configurations, which represent usage or consistency

Tools: objects that operate on user information. There are application tools such as editors and simulators, desktop tools such as browsers and workflow editors, system tools for managing operating systems, and hardware tools such as printers.

Tasks: objects that express the context in which tools process information. Tasks are instances of workflows bound to instances of information. Users are guided through tasks by workflow rules that determine the next operation, or operations, to execute based on interpretation of task context and state.

Access to user objects, in an arbitrarily large and heterogenous, multi-user environment, is restricted by security and concurrency mechanisms. The appearance and behavior is consistent regardless of the scale and complexity of the network containing, and facilities managing, user objects. For example, information objects which are distributed and have relationships that are being navigated or established across multiple heterogenous information management facilities, behave on the desktop as if they were managed by a single facility.

Desktop Management facilities support collaboration in formally defined projects and in informal transactions by managing user transactions that generate requests to appropriate information, task, and system management facilities. The facilities are responsible for:

- Defining user profiles, organization and project structures that specify and assign group and role membership used by facilities to determine resource access.
- Synchronization and concurrency mechanisms such as locking and versioning that facilitate and provide integrity for parallel activities.

Desktop Management facilities support the following functions:

- Installation and setup
- Session management
- Information management
- Tool management
- Task management

Installation and Setup

Installation and setup is required for for new installations, extensions, and modifications to the user's working environment. Such installations may include a network of heterogeneous facilities and platforms. Operating systems and facilities may be installed independently of this function; however, their existence and context is registered through interaction with this function.

The installation and setup process should be automatic except where information or choices are required. The dialogue with the system administrator executing this process must setting of environment variables, configuration files, and other internal information is the responsibility of this function based on information obtained from dialogue with the system administrator.

Session Management

The session management function is required for connection, customization, control, and recovery of the user's working environment, including:

- Login, which includes use of security services and the handling of exception and other events that have occurred since the last connection to establish and update the user's working environment.
- Logout, which includes support for handling uncommitted data and executing tools that are unable to continue execution while the user is not connected.

- System Configuration, which provides support for customizing and extending the user's working environment from an installed desktop.
- Event Logging, providing support for viewing and setting retention and usage policies.
- Undo/Redo, to support for reversing user transactions.
- Service functions for fault isolation, detection, and correction at the desktop.
- User, Group, and Role functions for creation, definition, and assignment.

Information Management

The information management function must support interfacility relationships, transactions, and notifications. Information management functions are presented in separate windows that inherit the behavior of these common functions. The desktop provides a consistent user interface to information management facility functions and assists with interfacility cooperation. The common functions are:

- Creation, destruction and concurrent usage (checkout/in) of objects.
- Browsing, navigation, and creation/destruction of object relationships.
- Functions for registering interest in object change notifications, setting and querying attributes, defining information types, and backup/restore.

Tool Management

Tool management functions provide the user interface for defining, registering, and using tools managed by tool management facilities. This includes:

- Editors for specifying tool characteristics and execution requirements.
- Methods for constructing and using collections of tools in toolboxes.
- Execution of selected tools with interpretation of conditions specifying information required, tool version, access, and execution environment.

Task Management

Task management functions provide the user interface for task creation, execution, control, and resource management. Creation involves binding an object to a workflow, or to a tool, with defaults presented based on information type. Execution, control, and scheduling functions include start, suspend, and resume task.

2.5.2 Related Standards and References

CAD Framework Initiative (CFI) Task and Session Model, May 2 1994.

ICL Common Facilities, Response to OMG Common Facilities Request For Information.

Inter-Framework Facility, Response to OMG Common Facilities Request For Information.

2.5.3 Relationship to Other Components of OMA

The Desktop Management Facility requires a protocol that supports cooperation between Task Management, Information Management and System Management Common Facilities for issuing requests based on desktop transactions.

The Desktop Management Facility may use Object Services and other Common Facilities.

2.5.4 Technical Issues

None

2.6 Scripting Facility

2.6.1 Description and Requirements

The Scripting Facility supports:

- A Turing-complete, interpretable language that supports functional decomposition. This is needed to support sending scripts as agents.
- Exposing of key interfaces to key Common Facilities, Object Services and ORB facilities at the language level.
- A visual programming environment (for example, keystrokes recording, mouse clicks recording) to create macros that can be executed by the language.

2.6.2 Related Standards and References

Typical examples of standards are:

Microsoft Visual Basic

Apple Script and Open Scripting Architecture

Scheme, CFI's macro language

TcL, Scripting language popular in the UNIX community

2.6.3 Relationship to Other Components of OMA

Script programs can:

- Be considered as agents in the Task Management Facility.
- Have hooks into the high level messaging service of Task Management Facility.
- Generate high level messages to the messaging service. Use services listed in the advertisement registry of CORBA.
- Send and receive events from the Event Service.
- Create, delete, copy, and move objects using the Life Cycle Service.
- Use the Collection and Relationship Services.

The script visual program environment may use the Application Development facility. In addition, names in the scripting language should be mappable to names in the Naming Service.

The Event, Life Cycle, Relationship, and Naming Services are described in *CORBA services*.

2.6.4 Technical Issues

Depending on the architecture chosen for scripting a common facility, there may be technical issues. If scripts are considered as agents, then the following issues need to be addressed:

- The set of scripting languages can expand from procedural languages to include declarative languages (rules, logic, and so forth).
- High level messages in task management can themselves be scripts. Therefore, the messaging facility needs to support sending of interpretable programs within it.
- The rules language of task management and the scripting language can be one and the same.
- Management tools for information storage and retrieval need to be able to store and retrieve scripts.
- Information exchange needs to support the exchange of scripts.