UNISYS

Cool ICE®

USERS GUIDE

RELEASE 1.0

March 1997
Printed in USA
Priced Item

UNISYS Cool ICE

Users Guide

Release 1.0

March 1997
Distribution Codes MU11G, RVRC, MU1V, MU1X
Printed in US America

Priced Item

78460847-000

The names, places, and/or events used in this publication are not intended to correspond to any individual, group, or association existing, living, or otherwise. Any similarity or likeness of the names, places, and/or events with the names of any individual, living or otherwise, or that of any group or association is purely coincidental and unintentional.

NO WARRANTIES OF ANY NATURE ARE EXTENDED BY THIS DOCUMENT. Any product or related information described herein is only furnished pursuant and subject to the terms and conditions of a duly executed agreement to purchase or lease equipment or to license software. The only warranties made by Unisys, if any, with respect to the products described in this document are set forth in such agreement. Unisys cannot accept any financial or other responsibility that may be the result of your use of the information in this document or software material, including direct, special, or consequential damages.

You should be very careful to ensure that the use of this information and/or software material complies with the laws, rules, and regulations of the jurisdictions with respect to which it is used.

The information contained herein is subject to change without notice. Revisions may be issued to advise of such changes and/or additions.

RESTRICTED – Use, reproduction, or disclosure is restricted by DFARS 252.227–7013 and 252.211–7015/FAR 52.227–14 & 52.227-19 for commercial computer software.

Correspondence regarding this publication should be forwarded to Unisys Corporation either by using the Business Reply Mail form at the back of this document or by addressing remarks to Product Information, Australian Centre for Unisys Software, 115–117 Wicks Road, North Ryde, NSW 2113, Australia.

About This Guide

Purpose

The Cool ICE Users Guide provides an overview of the Cool ICE environment. It also provides step-by-step information on using Cool ICE Administration to create and deploy script-based services.

Scope

Information in this document provides sufficient high-level information about the product, so that the reader should understand what it does and how its components fit together. It describes, using graphics, the various components of the Cool ICE environment, including the structure of a service, as well as some concepts that may be new to the reader.

In addition, it provides step-by-step instructions for the major tasks.

Audience

The audience falls into two main categories:

- Management who are looking for a means of providing Web-based services to their clients and staff, but who will not necessarily be involved in developing the product.
- Developers who have scripting experience and who need technical details about Cool ICE for developing and deploying services. These people will be technically knowledgeable and primarily interested in quickly getting started with Cool ICE.

Prerequisites

Developers wishing to use Cool ICE to develop dynamic services will need to understand the Cool ICE scripting language.

How to Use This Guide

Use Section One to get an overview of Cool ICE and the service development process. Use Section Two for step by step instructions on using the Cool ICE Administration module. The information in this section can also be found in the Cool ICE on-line help. The Appendix provides more technical details about services.

Organization

This user guide consists of the following sections:

Section 1. An Overview of Cool ICE

This section introduces you to Cool ICE, describes the Cool ICE deployment and development environment, and then takes you through the development process.

Section 2. Using Cool ICE

This section shows you how to use the Cool ICE Administration options to create and maintain categories and services, maintain a Style Guide, view event logs, and manage security.

Appendix A. The Structure of a Service

This describes the different parts of a service, and provides some information on specifying HTML hyperlink parameters for a service.

Contents

Figures

Section 1
An Overview of Cool ICE

Introducing Cool ICE

Welcome to Cool ICE, a complete environment for developing, running and managing Internet and Intranet solutions. Its robustness and scaleability allows you to take advantage of the World Wide Web for electronic commerce in a secure and flexible manner. Cool ICE has everything you need to quickly and easily create ***dynamic*** HTML pages.

Using Cool ICE, companies can provide their clients with new and better services. The areas where Cool ICE can be applied are unlimited: create new Internet-based systems for capturing customers, provide catalog services, take orders and bookings, provide management information/reporting capabilities and so on. It can change the relationship between suppliers and customers by providing on-line services over the Internet. Cool ICE will also help you breathe new life into existing applications, thus expanding the value of otherwise functionally acceptable applications.

Enormous potential for gaining a competitive edge is locked away in existing databases. With the combination of its powerful development capabilities and its ability to access multiple distributed databases, Cool ICE helps unlock and use this information to:

- rapidly deploy new and differentiating services
- improve customer service
- reach new markets
- reduce costs in manufacturing, distribution and support

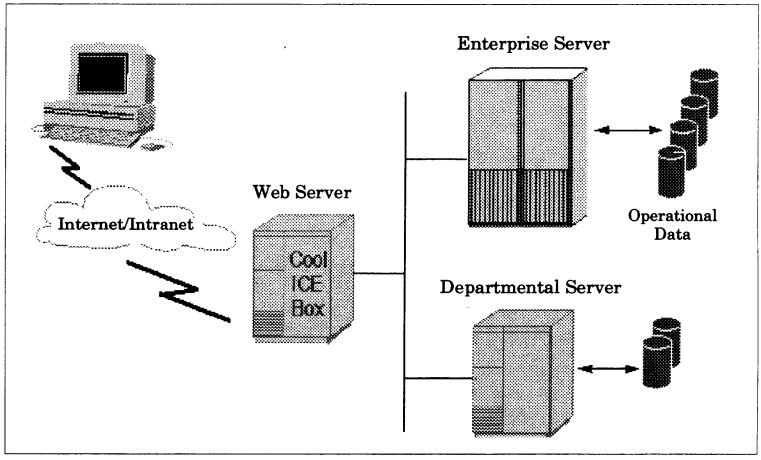


Figure 1–1 The Cool ICE Environment

Cool ICE and the Internet

The Internet is a computer network that connects millions of computers globally and provides world-wide communications to businesses and homes. An *Intranet* is based on Internet technology but exists only within an organization protected from unauthorized access.

Users are attracted to the Internet because it is easy to use and because it combines graphics, text, sound and animation into a rich communication medium.

Every computer program that communicates on the Internet is either a server or a client. A server offers a service to other machines on the network and a client requests a service from a server. On the client machine there will be a Web browser and on the server there will be Web Server software.

A browser is a program that navigates the Web and displays pages. The browser requests a page from a server based on the page's Internet address. It receives and displays pages that combine text, pictures, forms, sound, animation and hypertext links called hyperlinks.

A Web server stores pages and sends them to a browser when requested. It may also run programs referred to as Common Gateway Interface (CGI) scripts. With Cool Ice there is no need for CGI programming as the interface to the Web Server is completely transparent to the Cool ICE service developers. The interface to the Web Server is handled by the Cool ICE Gateway.

Pages

The basic document of the Web is a page. Pages are written in a language called HTML (HyperText Markup Language). A HTML page contains text along with tags, which are embedded commands that supply information about the page's structure, appearance and contents.

In a Cool ICE environment, HTML pages can be static pages or dynamically generated pages. A dynamic page is the output created by a Cool ICE Web service.

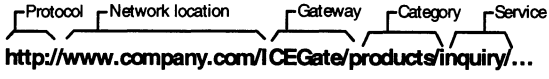
Hyperlinks

A hyperlink is a connection from a Web page to another file on the Web. The destination of the hyperlink can be another page, a multimedia file or a program. Hyperlinks are usually short pieces of text. A browser emphasizes hyperlinks by underlying and displaying them in a specific color. When a user clicks on a hyperlink, the browser passes the request off to the appropriate Web Server and waits for a page to be returned.

A Cool ICE Web service can be referenced and requested via hyperlinks. In Cool ICE, hyperlinks are often used for displaying a menu of different services and for drilling down for more specific details. The destination of the hyperlink is encoded as a Uniform Resource Locator (URL).

URLs

A URL gives the address of a file or service on the Web. The components of the URL are as follows:



- **Protocol** specifies the Internet service that will handle the request. This will always be HTTP for URLs addressing Cool ICE Web services.
- **Network location** is a unique name or TCP/IP address that identifies a Web Server.
- **Gateway** is (usually) the path identifying the folder containing a file. When addressing a Cool ICE Web service, this is the name of the gateway that provides the interface between the Web Server and Cool ICE.

- **Category** is the path into the Cool ICE Repository identifying the category in which to look for the service.
- **Service** identifies the name of the Cool ICE Web service.
- **...** indicates that one or more parameters (depending on the service) can be specified following the service name. These items are passed as input parameters to the service.

The network location and the gateway are provided by the Cool ICE environment at runtime; therefore services do not require any changes if they are moved to another Web Server.

Designing Web Services

When designing services for the Internet, the nature of the Internet as a true client/server environment must be taken into consideration. The architecture and design of Web services is much different from a more traditional dialog-based environment. Every service request from a browser is treated as a new transaction independent of what was previously requested.

For example, consider a user dialog where the user receives a blank form, fills in the form and sends the form back, where it is then inserted in the database. This actually requires two interactions with the user. First, the user requests the form and secondly, the user sends the form back. In a traditional dialog-based environment this would often be designed as one service handling the two interactions with the user. In an Internet environment this would normally be designed as two separate service transactions. One service displays the blank form while another service handles the input, validation, database update and possibly returns a confirmation.

Maintaining State

Unlike normal applications, Web transactions are 'stateless'. This means that all requests to a Web server are treated as completely separate transactions. After each transaction, the state of the transaction is lost.

Cool ICE, however, provides a mechanism for maintaining a session with the user. The traditional way of remembering and passing information from one dialog to the next using hidden fields in forms and attaching parameters to URLs is a technique that will be sufficient in most cases. In addition to this, Cool ICE is able to keep track of users and the various services and pages they are requesting without requiring them to provide their user ID and password each time. It does this by maintaining a Session ID. This session number is allocated the first time a user requests a service, and is

returned to the Web Service Handler each time a hyperlink is followed or a page returned, allowing Cool ICE to track users and their security profile. If the data is to be stored on a remote server, the security profile is passed to that service.

Requirements

You will need a workstation and a Web Server in order to develop and deploy Cool ICE services.

Workstation

Cool ICE requires the following software on your workstation:

- Windows 95 or Windows NT Workstation 4.0.
- Software for connecting to Cool ICE, as follows:
  - MAPPER Presentation Client (MPC) or
  - Power Client Edition (PCE) or
  - MAPPER System for Windows (MSW)
- A HTML authoring tool (such as Microsoft FrontPage).

Web Server

Cool ICE supports the following Web Servers:

- Netscape Enterprise Server.
- Microsoft IIS.

Cool ICE requires the following software on your Web Server:

- Windows NT Server 4.0.
- MAPPER-NT 5.3.1 or higher. The components of the Cool ICE product run under the Unisys MAPPER Rapid Application Development shell.
- Cool ICE 1.0.
- Cool ICE Graphics. This is included with Cool ICE and allows you to create business graphics (such as bar charts, pie charts and so on) dynamically.

Inside Cool ICE

Cool ICE Components

Cool ICE consists of the following components:

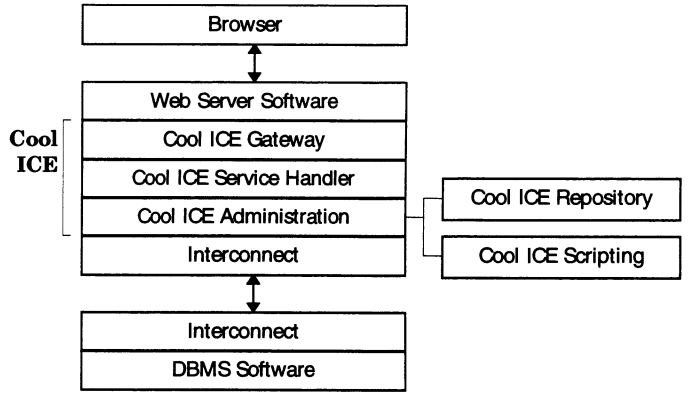


Figure 1–2 Cool ICE Components

Cool ICE Gateway

The Cool ICE Gateway is an interface that provides a connection between the Web Server and the Cool ICE Service Handler.

Cool ICE Service Handler

The Cool ICE Service Handler is the runtime module that handles requests for services coming from the client browser over the Internet/Intranet. It uses the Repository for information about the Web services and performs the service. The Web service developers do not need to know CGI or API programming as the interface to and from the Web Server is completely handled by the Cool ICE environment. The developers can concentrate on the business logic.

The following is an overview of the features provided by the Cool ICE Service Handler:

Gateway Transparency. The interface to the ICE Gateway and Web Server is transparent to the service. The service developer does not need to be concerned about

how to communicate with the Web Server through the gateway. *Input* from a browser is delivered to the services in a way that is suitable for the ICE scripting language and is easy to handle for the service developer. *Output* to the browser is taken from the service and handed over to the gateway.

Location Transparency. A service can be located in the Cool ICE Repository on the local Web Server or on a remote server. The physical location of the service is transparent to the service developer. You can move or copy a service from one location to another without having to modify it.

Service Security. Extensive application level security is provided by the Cool ICE Service Handler. Before a service request from a browser is performed, the Service Handler will verify that the user has the necessary rights to request the service. If a service has been allocated a security profile, the Service Handler will check that the user has a matching profile.

Service Monitoring. The Cool ICE Service Handler monitors all events. It logs errors that occurred in the Web services, as well as all user activity so that it is possible to get useful information about how the Web services are being used.

Error Logging. The Service Handler maintains an error log of services that fail. This helps the service developer to analyze and solve the problem.

Tracing and Debugging. Tracing and debugging a service in a client/server environment such as the Internet is not an easy task, because a service is performed in the background on the server and debugging tools are normally not available. The ICE Service Handler provides a mechanism to capture input from a browser, then allows the service to be performed in the foreground using normal debugging techniques (such as displaying information and stepping through a service).

Style Guide. The Cool ICE Style Guide allow you to provide a consistent look and feel to HTML documents generated by a dynamic service, and to be able to change this at one source. They also provide an easy way of maintaining common variables to be used in dynamically generated HTML code. This provides a single point of control for shared variables. A variable just needs to be changed in one place for it to have effect at run time in all services referencing it.

Automatic Menu Generation. A menu of categories and services will automatically be generated as a default when a browser comes into Cool ICE without requesting a particular service. The menu will only contain services specific to that user, according to the user's security profile.

Cool ICE Administration

The Cool ICE Administration module (shown below) is a graphical interface to Cool ICE. It allows the service developer to create, manage, distribute and monitor the static and dynamic services stored in the Cool ICE Repository.

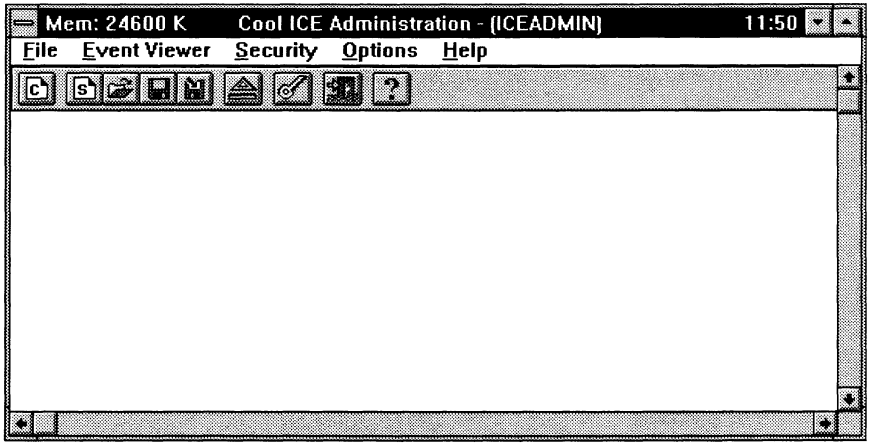


Figure 1–3 The Cool ICE Administration Environment

Cool ICE Administration options also allow you to carry out the following tasks:

- manage security
- view log information
- maintain Style Guides
- configure Web server directories
- maintain aliases
- configure sign-on settings
- manage images

Cool ICE Service Repository

Cool ICE uses a central Repository to store information about services, such as their attributes and location. While the Repository is always stored on the local Web Server, services may be physically distributed on one or more servers. A service is executed where it physically resides. (Sometimes it is easier to execute a service if it resides on the server that holds the data it requires). The Repository keeps track of which services are currently open and to whom.

Interconnect

Mechanisms for accessing data and applications depend on the nature of the service (for example, file transfer, SQL and so on). Database management systems can be accessed in native mode. Any database that supports Microsoft's ODBC standard can also be accessed.

Cool ICE Scripting

You develop a dynamic Web service using the Cool ICE scripting language. This is a powerful server-side scripting language that allows for fast development of the business logic of Web services. Database access to ODBC compliant databases and enterprise databases such as DMS 2200, RDMS 2200, DMS II and DB2 is provided through SQL. Knowledge of HTML is not essential as the HTML output can be created as a template using an HTML authoring tool.

What is a Service

Service is the name used by Cool ICE for any service provided by a company via the Internet or a corporate Intranet. Cool ICE distinguishes between static and dynamic services:

- Static service. A static service allows a user to access documents that do not include added script. These are called static documents. They usually consist of HTML code and can be developed using any standard Web authoring tool. Once created, they may be registered for access to designated users.
- Dynamic service. A dynamic service generates content and HTML code at the time of invocation. It may access a local or commodity database, manage inquiry forms and so on. Dynamic services can be developed by adding script to an HTML page, or they can be built from scratch. They can include complex application logic, access to multiple databases, graphic art, dynamic business

graphs and static text. A dynamic service can be triggered as a result of an end-user action, thus providing a true transaction environment on the Web.

Service Categories

Services are grouped into categories. A category may group a set of services that form an application or that form some other relationship; it is up to the developer as to how they wish to categorize services.

A category can be one of the following:

- Local category. The services for the category are stored on the local server.
- Remote category. The services for the category are stored on a remote server.

Information for both these categories is kept in the Repository on the local server. The Cool ICE Administration module handles the required networking for storing and retrieving services in a remote category.

Requesting a Service

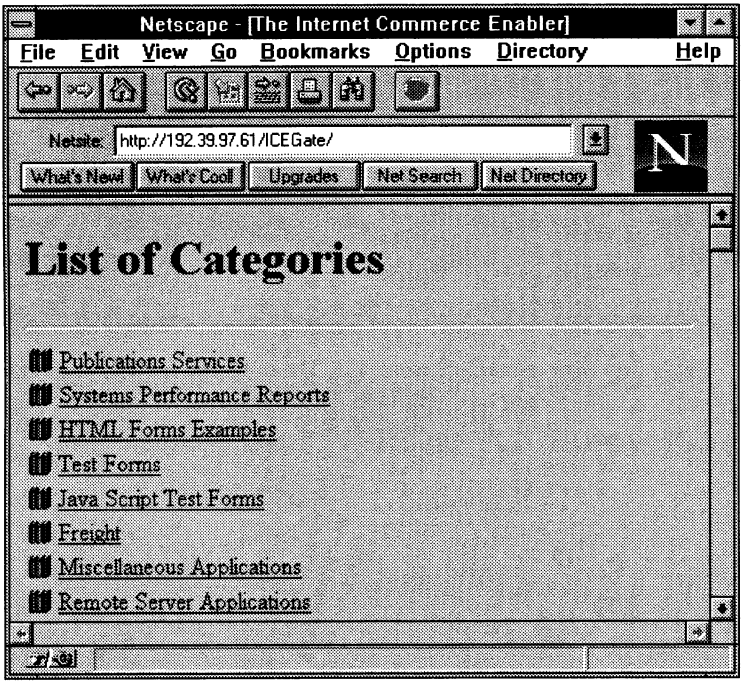
The following shows the steps involved in requesting a Cool ICE service to locate a particular marketing document on your companies Intranet:

1. Within their browser, the user enters the URL for the Cool ICE Web Server. This has the following format:

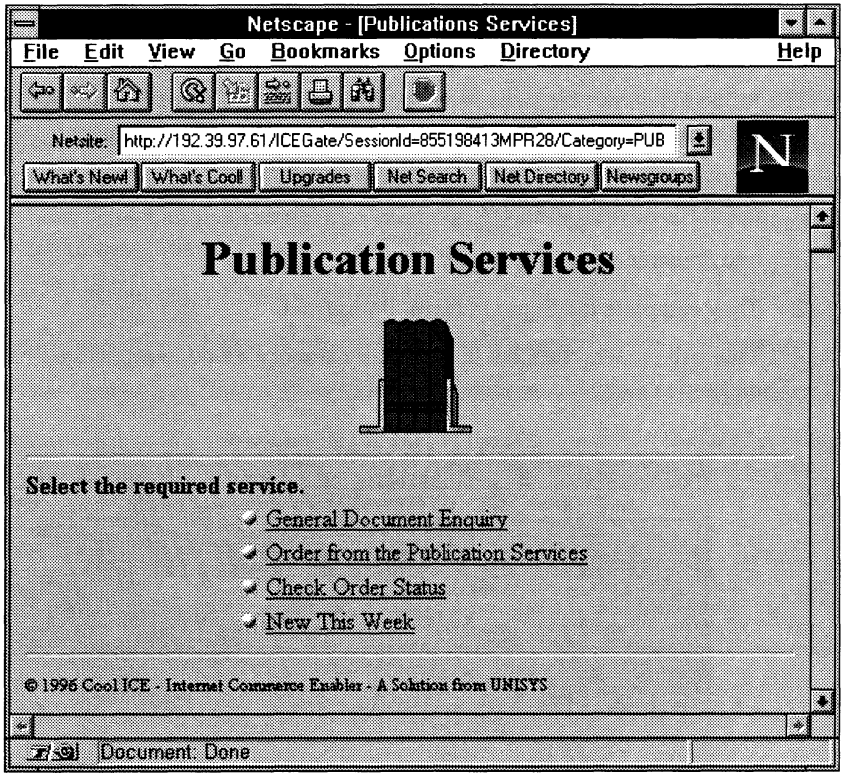
   http://machine-name/ICEGate/

2. They complete the Sign-On form, if required.

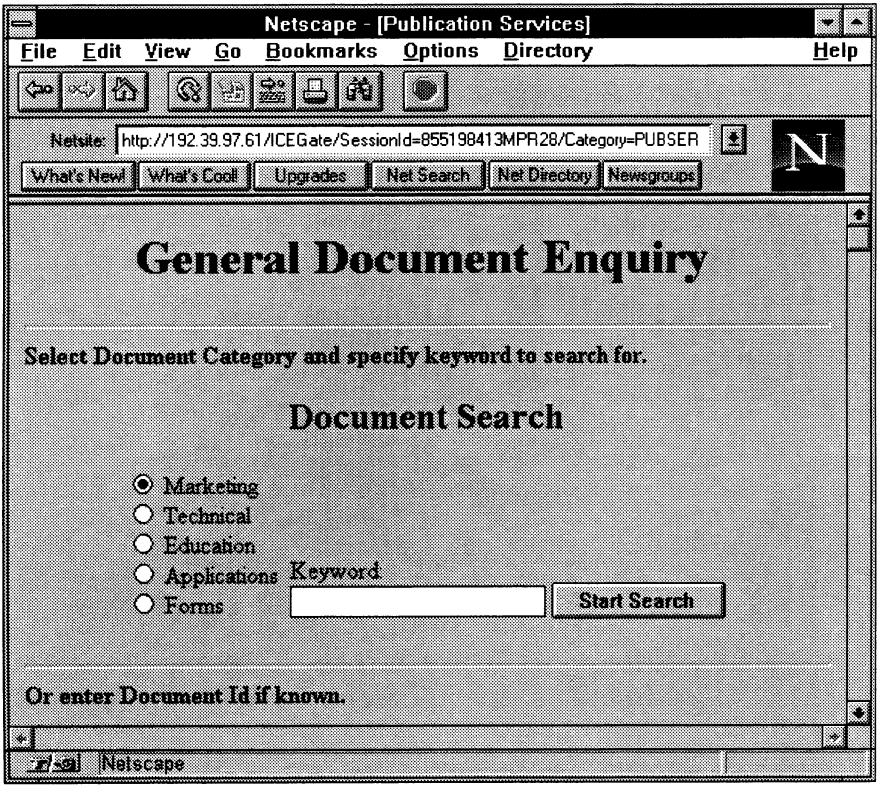
3. This will present the Cool ICE default category menu, containing a list of categories available on their Web Server.



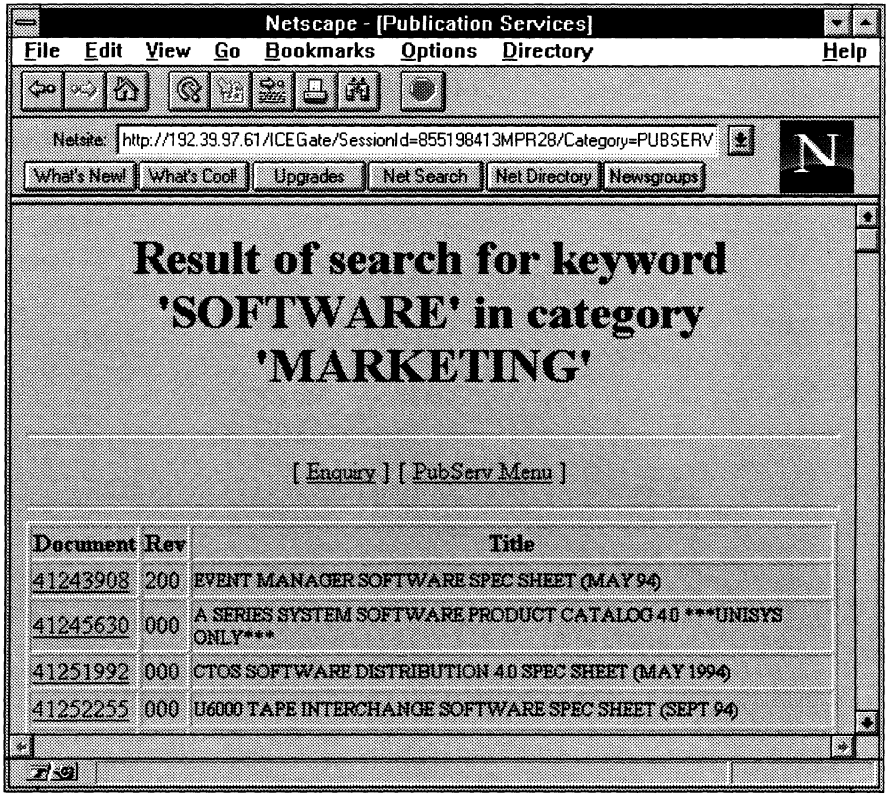
4. From this menu they can select a category. The services for the category will then be listed. For example, selecting the Publications Service category in the previous screen, would display the following Web page:



5. From this they can select a service. The dynamic HTML page generated by this service would then be displayed. For example, selecting the General Document Enquiry service in the previous screen, would display the following Web page:



6. This service allows them to enter a search word. For example, entering the word "software" in the search field in the screen above and pressing the Start Search button, would display the following dynamically generated Web page:


Netscape - [Publication Services]
File Edit View Go Bookmarks Options Directory Help
Netsite: http://192.39.97.61/ICEGate/SessionId=855198413MPR28/Category=PUBSERV
What's New! What's Cool! Upgrades Net Search Net Directory Newsgroups
Result of search for keyword 'SOFTWARE' in category 'MARKETING'
[Enquiry] [PubServ Menu]
Document Rev Title
41243908 200 EVENT MANAGER SOFTWARE SPEC SHEET (MAY 94)
41245630 000 A SERIES SYSTEM SOFTWARE PRODUCT CATALOG 4.0 ***UNISYS ONLY***
41251992 000 CTOS SOFTWARE DISTRIBUTION 4.0 SPEC SHEET (MAY 1994)
41252255 000 U6000 TAPE INTERCHANGE SOFTWARE SPEC SHEET (SEPT 94)

What Happens When You Request a Service

The following is the sequence of steps that take place when a request is made via a browser for a service (see Figure 1–4 following).

1. A service is requested by a user through their Web browser. The request may be made by the user clicking on a hypertext link, pressing a button or by typing in the URL directly (see "Specifying Hyperlinks" in Appendix A.). This request is passed to the Web Server.
2. The Web Server interprets the request and calls the Cool ICE Gateway specified in the URL.
3. The Cool ICE Gateway creates a browser input file. This contains hyperlink information, form input and so on, depending on the nature of the request. It then calls the ICE Service Handler.
4. The Service Handler uses the browser input file to match the request with a service in the Repository. It determines whether the user is authorized to access the service; if not then they are denied access.
5. The Service Handler then gets the service location and attributes from the Repository and creates the input parameters for the service in a service input report. The report contains,
   - Global Cool ICE system information.
   - Style Guide information.
   - Browser input.
6. Input parameters from the service input report are loaded into the service, and the service is executed. The service may then retrieve the Web page information, either from a static source or by creating it dynamically. The retrieved data is then formatted into HTML code and passed to the Service Handler.
7. The Service Handler writes the HTML document to a browser output file and returns control to the Cool ICE Gateway.
8. The Cool ICE Gateway returns control to the Web Server.
9. The HTML document is sent to the browser where it is displayed.

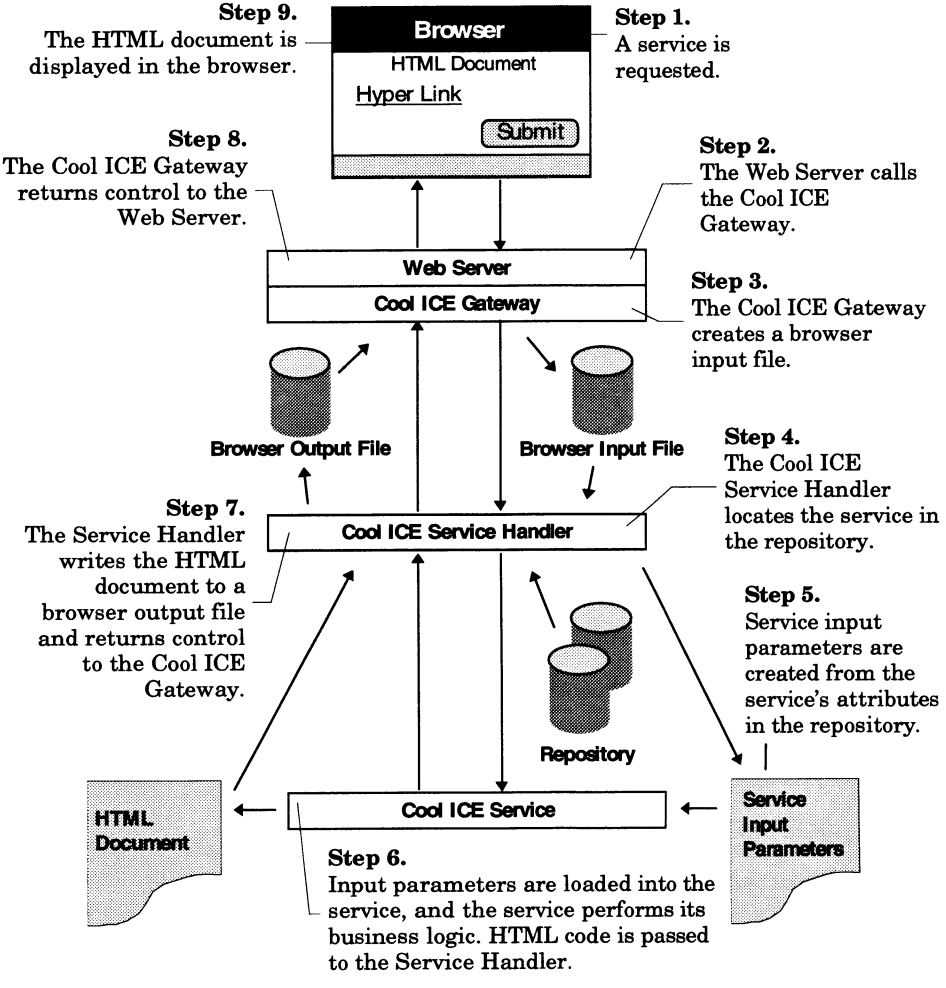


Figure 1–4 Requesting a Service

The Development Process

Creating a Dynamic Service

This section describes the steps required to create a dynamic service (see Figure 1–5 following).

1. Create an HTML document using an HTML authoring tool of choice (for example, Microsoft FrontPage).
2. Save the HTML file as a normal .HTM file. This will be used as a *template* for the service.
3. Invoke the Cool ICE Administration module.
4. Create a new dynamic service based on the HTML template you created in Step 2.
5. Import the HTML template. This automatically adds sequences of script to the beginning and end of the HTML code, and uploads any required images onto the Web Server.
6. Edit the service as required by inserting any additional script. You will need to be familiar with Cool ICE scripting to do this.
7. Save the service within a category in the Cool ICE Repository. The service is now available for testing via a browser.
8. From within your browser enter the URL for the required service. This URL has the following format:

   http://machine-name/ICEGate/ Category/Service

   If you do not know the full path to the service, you can specify the URL for the Cool ICE Web Server instead, as follows:

   http://machine-name/ICEGate/

   This will present a menu of available categories. Selecting a category from this menu will display a menu of services for the category, from which you can select your service.
9. The HTML page for that service will now be dynamically generated and displayed in your browser (see "Requesting a Service" on page 1–11).
10. If you need to enhance the HTML document you can export it (from within the Cool ICE Administration module) to a directory.

11. You can now use the HTML authoring tool to edit the HTML template. Having made your edits you can import the modified HTML template back into your service.

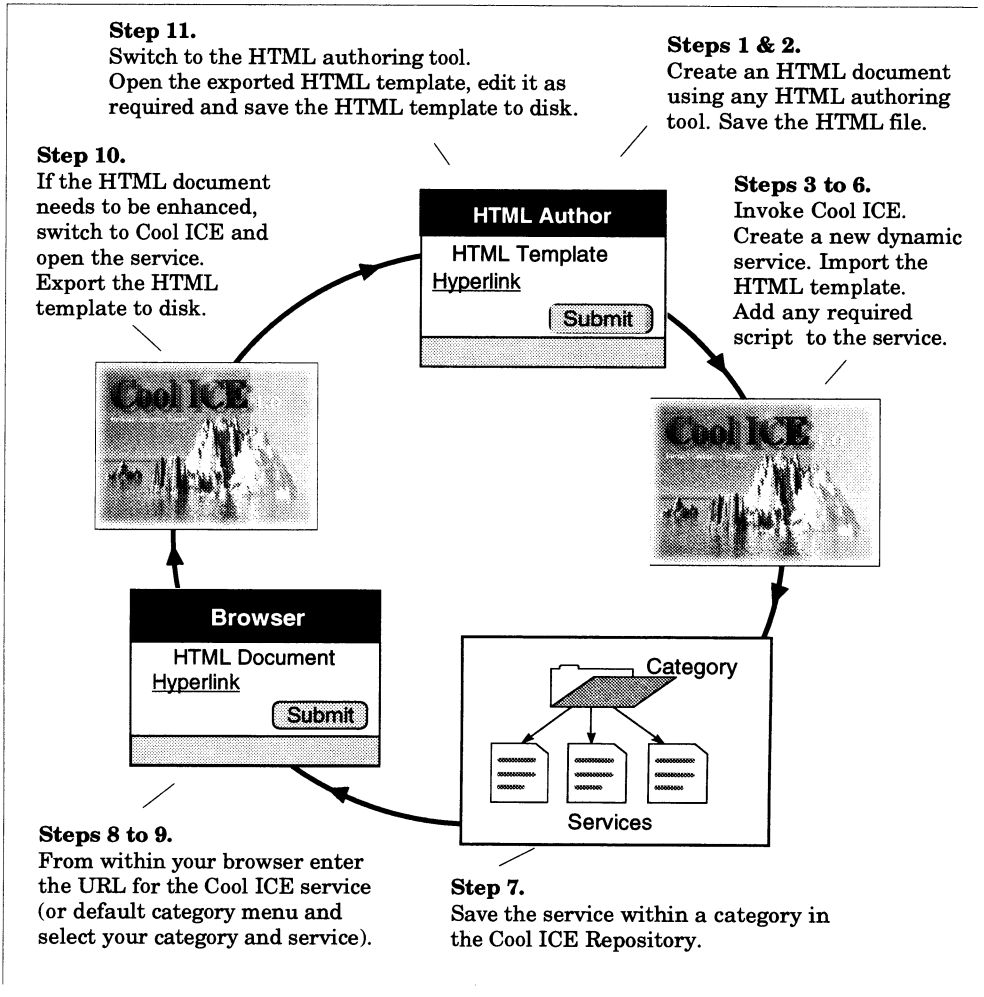
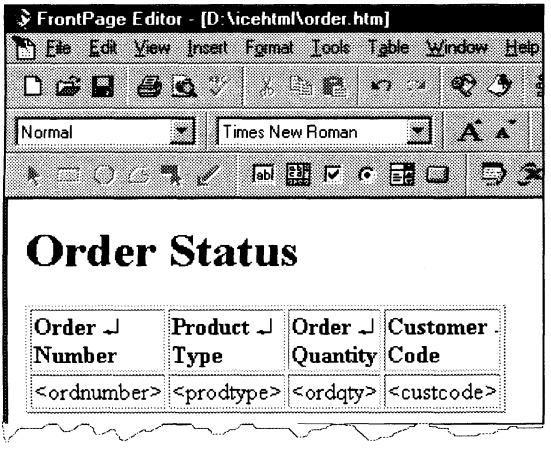


Figure 1–5 The Service Development Process

An Example of Creating a Dynamic Service

The following example takes you through the creation of an order-status service that displays a list of orders by customer code:

1. The following shows the order-status Web page being created in Microsoft FrontPage:



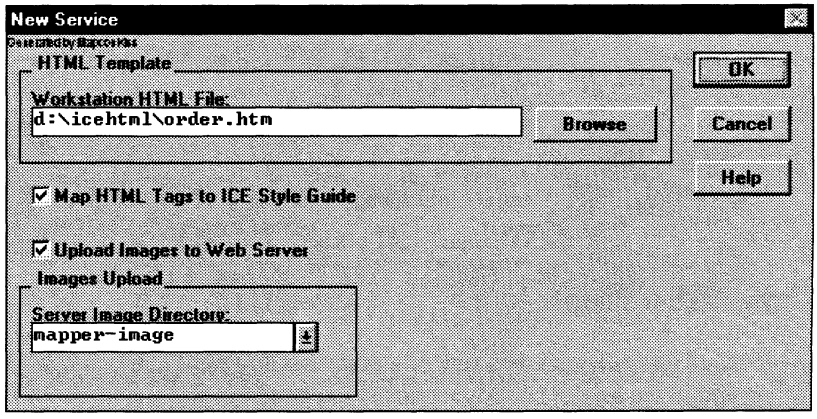
Note that only one data row of a customer order has been created. The script within the service will dynamically generate the actual number of rows depending on the number of order lines. The variable fields will be filled in with data from the customer-order database.

2. The following shows part of the HTML code *automatically* generated for the order-status page by the Microsoft FrontPage authoring tool:

```
<body BGCOLOR=#C0C0C0 TEXT=#000000>
<h1 align="left">Order Status</h1>
<table border="1" BGCOLOR=#C0DCC0>
<tr>
    <td><strong>Order<br>Number</strong></td>
    <td><strong>Product<br>Type</strong></td>
    <td><strong>Order<br>Quantity</strong></td>
    <td><strong>Customer<br>Code</strong></td>
</tr>
<tr>
```

```
        <td>&lt;ordnumber&gt;</td>
        <td>&lt;prodtype&gt;</td>
        <td>&lt;ordqty&gt;</td>
        <td>&lt;custcode&gt;</td>
    </tr>

    </table>
```

3. This HTML code needs to be imported into a service. To create a dynamic service based on an HTML template, you select the *New Service* option from the Cool ICE Administration menu and select the *Dynamic Service Based on HTML Template* option. The following shows the New Service dialog:



In the Workstation HTML File field is the pathname to the HTML template (created in Step 1) to be used for this new service.

4. The following shows the HTML code after it has been imported:

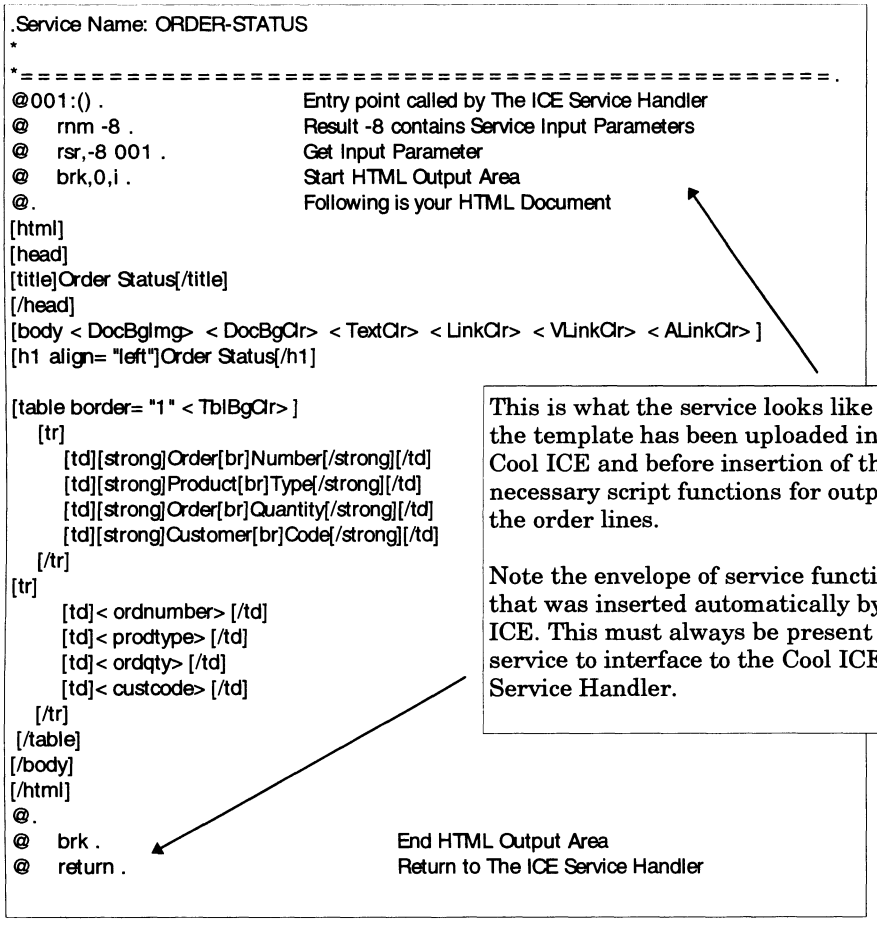


Figure 1–6 Imported HTML Code

Heading and footing script has been automatically added, and the HTML tag delimiters converted. This script is required to interface to the ICE Service Handler, and is explained in Appendix A, "The Structure of a Service".

Note: *The standard HTML tag delimiters < > have been converted to [ ]. This is in order to avoid conflict with the delimiters for script variable names. The ICE Service Handler will convert them back before passing the document over to the ICE Gateway.*

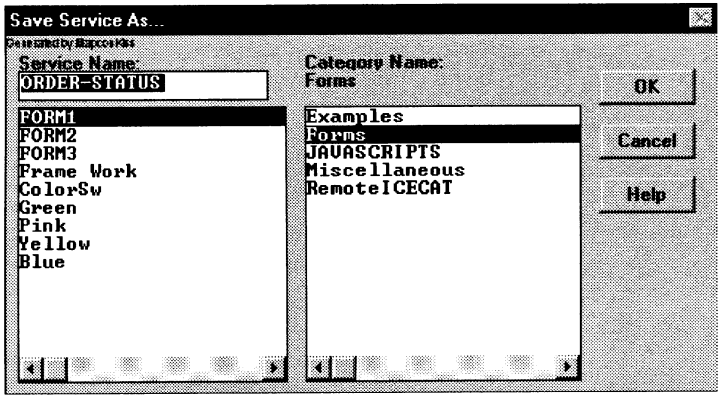
5. The following shows the HTML code after the insertion of script to access the customer-order table and to output one row for every order line:

```
.Service Name: ORDER-STATUS
*
*================================================
@001:() .            Entry point called by The ICE Service Handler
@   rnm -8 .         Result -8 contains Service Input Parameters
@   rsr,-8 001 .     Get Input Parameter
@   brk,0,i .        Start HTML Output Area
@.                   Following is your HTML Document
[html]

[head]
[title]Order Status[/title]
[/head]

[body < DocBgImg>  < DocBgClr>  < TextClr>  < LinkClr>  < VLinkClr>  < ALinkClr> ]

[h1 align= "left"]Order Status[/h1]

[table border= "1" < TblBgClr> ]
   [tr]
      [td][strong]Order[br]Number[/strong][/td]
      [td][strong]Product[br]Type[/strong][/td]
      [td][strong]Order[br]Quantity[/strong][/td]
      [td][strong]Customer[br]Code[/strong][/td]
   [/tr]
@   sor,0,D,1 " 'CustCode' ",1 .                    Sort by Cust Code
@   fdr,-0 " 2-1 '*',= rln 1-1 < lt> a .            Find start of data
@011:rln,,,019 'OrderNumber','ProductType','OrdQty','CustCode' \
          < ordnumber> a,< prodtype> a,< ordqty> a,< custcode> a .  Get a row
@   ldv,p < ordnumber> ,< prodtype> ,< ordqty> ,< custcode>  .      Pack variables
   [tr]
      [td]< ordnumber> [/td]
      [td]< prodtype> [/td]
      [td]< ordqty> [/td]
      [td]< custcode> [/td]
   [/tr]
@   gto 011 .                                    Go get next row
@019.                                            End of rows
[/table]
[/body]
[/html]
@.
@   brk .            End HTML Output Area
@   return .         Return to The ICE Service Handler
```

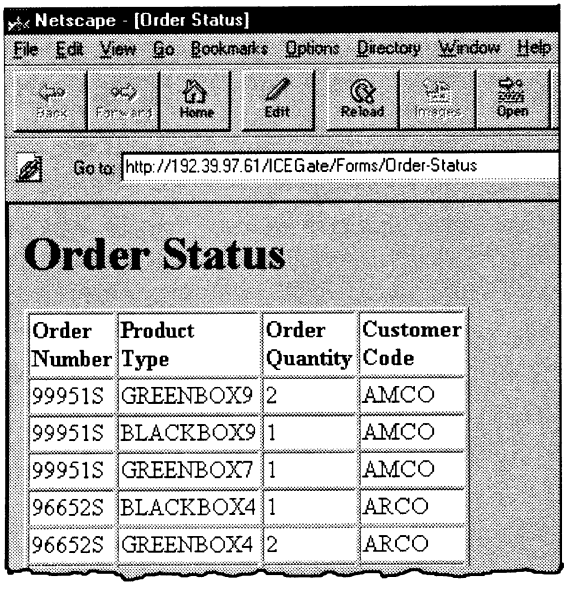
Additional script.

Figure 1–7 HTML Code Including Script

6. The service is now saved as an entry in the Forms category in the Cool ICE repository. The following shows the Save Service As dialog:



7. The following shows the result of the dynamic Web page for the service after it has been requested from the browser:



Note that the additional script added to the page has generated lines of data for each entry in the customer-order database. Any changes to the customer-order data will be reflected the next time the page is displayed.

Section 2
Using Cool ICE

This section shows you how to use the Cool ICE Administration options to create and maintain categories and services, maintain a Style Guide, view event logs, and manage security. It contains the following subsections:

- Creating and Maintaining Categories
- Creating and Maintaining Services
- Using a Style Guide
- Viewing Events
- Security
- Other Administration Options

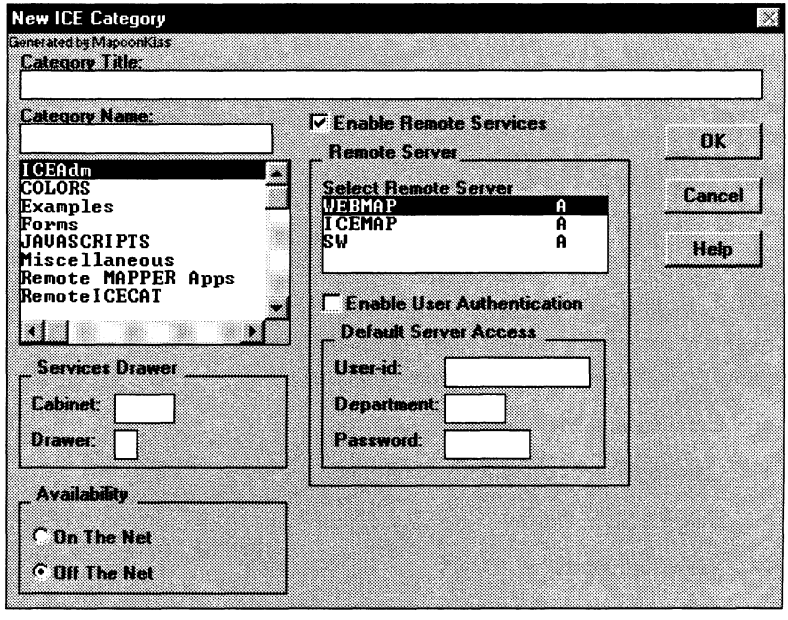
Creating and Maintaining Categories

A category is used for storing related services. Creating a category involves allocating a drawer in the system as the area for storing the services for the category.

On a local system this drawer is generated automatically. On a remote system the drawer must first be allocated and generated (as a free form 80 characters wide drawer) by the coordinator of the remote MAPPER system.

To create a category

1. From the Cool ICE Administration menu, choose *File*, *ICE Category* and then *New Category*. This displays the New ICE Category dialog.



2. In the Category Title field, enter the text that will be displayed for the category in the Cool ICE default category menu shown in the browser. Up to 70 characters can be entered.

3. In the Category Name field, enter a unique name for identifying the category in the Repository.

4. In the Services Drawer frame, enter the MAPPER cabinet and drawer to be allocated for storing the services within the category.

5. Complete the other fields on this form as required.

   - **Availability.** This allow you to take a category off the Internet. You might wish to do this while maintaining one or more of its services. By default, a new category is Off The Net. When you have added services to a category, you can put it On The Net using the *File*, *ICE Category* and then the *Category Properties* option.

   - **Enable Remote Services.** You need to check this if the category is to be located on a remote MAPPER system. When checked, a list of remote MAPPER systems (configured in the Networking Configuration Report) is displayed for selection. Cool ICE will verify that the cabinet and drawer specified for the Service Drawer field have been generated on the remote

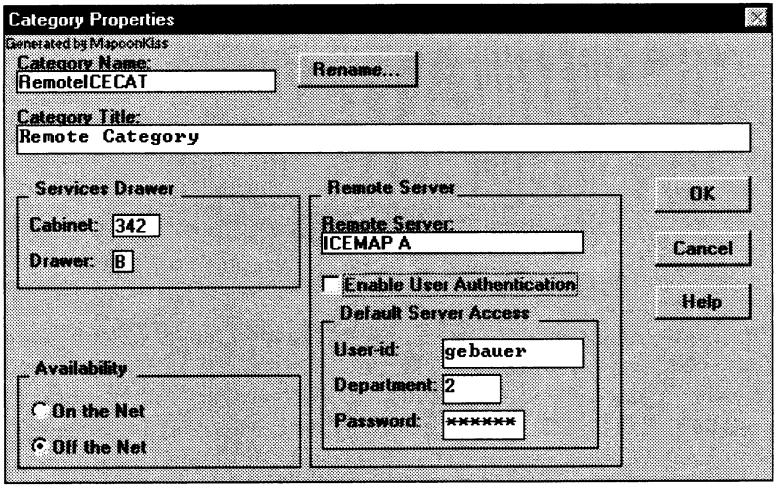
MAPPER system and the drawer is empty. The coordinator of the remote MAPPER system must have allocated and generated the drawer as a free form 80 character wide drawer.

- **Enable User Authentication.** This is displayed only if Enable Remote Services is checked. It ensures that for a user to access any service within this category, their user ID must be passed to the remote MAPPER system and be configured as being valid on the remote system. When unchecked, a default user ID, department and password can be specified here. These are then used for allowing access to the services within the category. This user authentication information must be defined on the remote system.

  Note: Enabling user authentication provides a high level of security but requires additional administration.

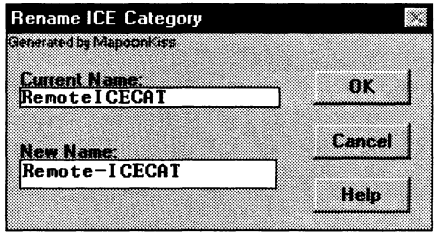
To modify a category

1. Choose *File*, *ICE Category*, and then the *Category Properties* option. This displays the Category Properties dialog.
2. Select the category you wish to modify. This displays the Category Properties dialog for this category.



3. Modify these fields as appropriate.

To change the displayed category name you need to click the Rename button. This displays the Rename ICE Category dialog, where you can enter a new name for the category in the Repository.

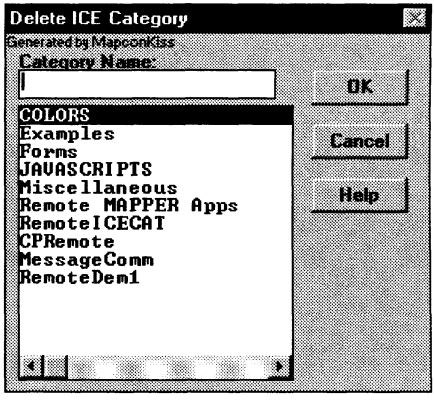


Cool ICE automatically changes the name everywhere it appears in the Repository. For local categories, the drawer name will also be changed where used by the MAPPER system.

The cabinet and drawer location cannot be changed. To move a category to another location, you must create a new category and use the *File*, *Copy Service* option to copy the service.

To delete a category

1. Choose *File*, *ICE Category* and then the *Delete Category* option. This displays the Delete ICE Category dialog.



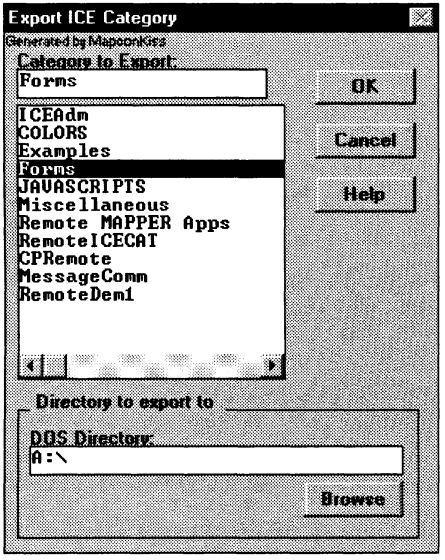
2. Select the category you wish to delete from the selection list and press OK. Repository information for both local and remote categories is deleted. However,

the MAPPER drawer containing the services is physically deleted only for local categories.

To export a category

A category and its services can be saved to another media. For example, you might wish to exchange a category with another Cool ICE system, or back up the service specifications for a category.

1. Choose *File*, *ICE Category*, and then the *Export Category* option. This displays the Export ICE Category dialog.



2. From the Category to Export selection list, select the category you wish to export.
3. In the DOS Directory field, enter the DOS directory where you wish to write the category and its services. This could be a directory on your hard drive on you local workstation. Or it could be a diskette on your workstation (for exchanging the category with another Cool ICE system). Cool ICE will export all the Repository information for the category and its services, and will export the service specifications themselves including any associated Style Guide information.

To import a category

This option allow you to import a previously exported category.

1. Choose *File*, *ICE Category*, *Import Category*. This displays the Import ICE Category dialog.
2. From the Category to Import into selection list, select the category you wish to import to. This should be an empty category (a category that does not contain any services).
3. In the DOS Directory field, enter the directory name to which the category was previously exported.

Cool ICE will import all services included in the category. If you wish to import selected services only, we recommend you import the category to a temporary category and use the *File*, *Copy Service* option to copy selected services from there.

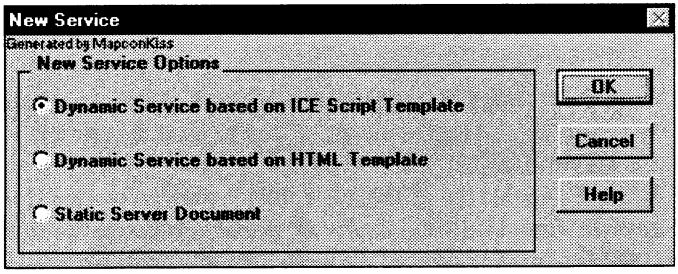
Creating and Maintaining Services

Three type of service can be created, as follows:

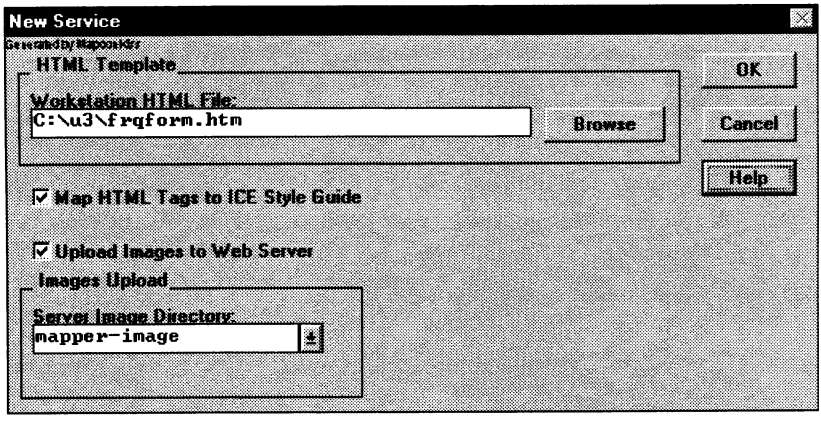
- **A dynamic service based on a Cool ICE script template.** A Cool ICE script template includes the essential script for developing a dynamic service. Once created, you can add to this script as required.
- **A dynamic service based on an HTML template.** An HTML template created using a standard HTML editing tool is imported into the service, and the necessary script to run the service automatically added. You can then add your own script to the service. This option is useful where you wish to create inquiry forms, input forms and output tables, which can be more easily created using an HTML authoring tool.
- **A static server document.** This allows you to register a static document in the Cool ICE Repository. The document is assumed to be an HTML document. For the document to be available to the Internet/Intranet, it must reside on the Internet/Intranet server. Using this option ensures that Cool ICE automatically handles the transfer of the document from the local workstation to the Internet/Intranet server.

To create a service based on an HTML template

1. Create the HTML file in an HTML editor and save it.
2. Choose *File*, *New Service*. This displays three New Service Options.



3. Select the second option: Dynamic Service based on HTML Template. This displays the New Service dialog.



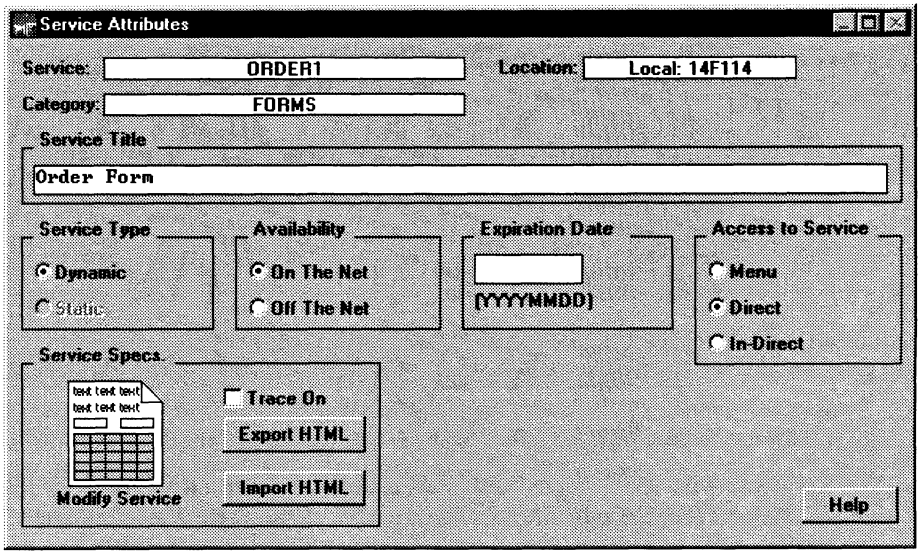
4. In the Workstation HTML File field, specify the path to the HTML file.
5. If you wish to use Style Guide information for this service, check the Map HTML Tags to ICE Style Guide checkbox. See "Using a Style Guide" on page 2–13.
6. If the HTML template references images, check the Upload Images to Web Server checkbox. This ensures that these images are stored in the directory specified in the Server Image Directory selection list. Cool ICE tries to determine the location of the image based on its reference specified in its HTML <IMG> tag. If the image cannot be found (because it is not accessible from your workstation), you will be asked to specify the exact location or to skip the uploading of the image.

   Before an image is uploaded to the server, Cool ICE will check if the image file already exists on the server and if so you will be asked to confirm replacing the image.

7. Press OK. The HTML file and any images are imported into the Cool ICE server as a dynamic service.

   When the HTML file is uploaded, sequences of script are automatically added to the beginning and end of the code. This script is required to interface to the Cool ICE Service Handler, and is explained in "The Structure of a Service" on page 1–21. Note also that the HTML tag delimiters are converted from angle brackets <> to square brackets []. This is done to avoid conflict with the delimiters used by MAPPER variable names. The Cool ICE Service Handler will convert them back to proper tag delimiters before passing the document over to the Cool ICE Gateway.

   The Services Attributes dialog is now displayed.



   The attributes for a service are stored in the Cool ICE Repository together with the specifications for the service. The attributes are displayed in the Service Attributes window when you are either creating a new dynamic service or opening a service.

8. In the Service Title field, enter the name to be used for the service when appearing in the Cool ICE default category menu displayed in the browser.

9. Complete the other fields on the dialog as appropriate.

The Service and Category fields are specified when you save the service.

The Availability field allows you to make the service available on the Internet or to take it off the Internet for maintaining.

The Access to Service field identifies how the end user can access the service, as follows:

- **Menu**. Indicates that you wish to use this service for displaying a menu of services available within the service's category. In other words, you will need to create a service that contains hyperlinks to other services in the same category. When Menu is not selected, Cool ICE generates a list of available services within the Cool ICE default category menu.
- **Direct**. Indicates that you wish to include a hyperlink to this service in the Cool ICE default category menu.
- **In-Direct**. Indicates that the service will not be included in the Cool ICE default category menu. Access to the service can only be made indirectly from other services.

10. Add any script required by the service by clicking on the Modify Service document icon. This displays the service specification. You need to be familiar with MAPPER scripting to add script.
11. When you are finished, save the service. Choose *File*, *Save Service* to display the Save Service As... dialog, and enter the category and service details as appropriate.

To create a static server document

When you create a service based on an HTML static document, the Service Attributes dialog will display an additional Static Server Document frame.

Enter here the location of the HTML document on the workstation, the directory on the server to which the document is to be transferred and the name of the file within the document directory.

When Transfer to Server? is checked, Cool ICE will automatically transfer the document from the local workstation to the server. When the document is updated and a new version needs to be transferred to the server, all that is required is for the service to be opened and the Transfer to Server? feature checked.

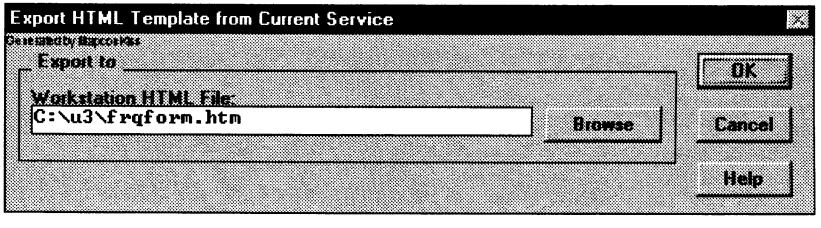
Importing and Exporting HTML

You can export the HTML template for a service in order to edit it using an HTML editor. When edited, you can import it back into the service. Exporting and importing is done using the Export HTML and Import HTML buttons on the Service Attributes dialog, as follows:

- Use the Export HTML option when you want to maintain and update the HTML template within a service. The HTML section of the service will be exported to a file on your workstation. This file can then be opened in the authoring tool and updated. When finished, save the template to the file ready to be imported back into the service.
- Use the Import HTML option to import a modified HTML template, replacing the template within the current open service. Use this option after you previously exported the HTML template from a service and updated the template. The HTML template being imported will replace everything within the current open service between the start and end HTML tags ([HTML] and [/HTML]).

To export an HTML file

1. Open the service using the *Open Service* option.
2. Click the Export HTML button on the Service Attributes dialog. This displays the Export HTML Template from Current Service dialog.



3. In the Workstation HTML File field, enter a file name for the HTML code. This will usually default to the original location of the HTML file. You can overwrite this file name if required. You can now edit the HTML code as necessary using any HTML authoring tool.

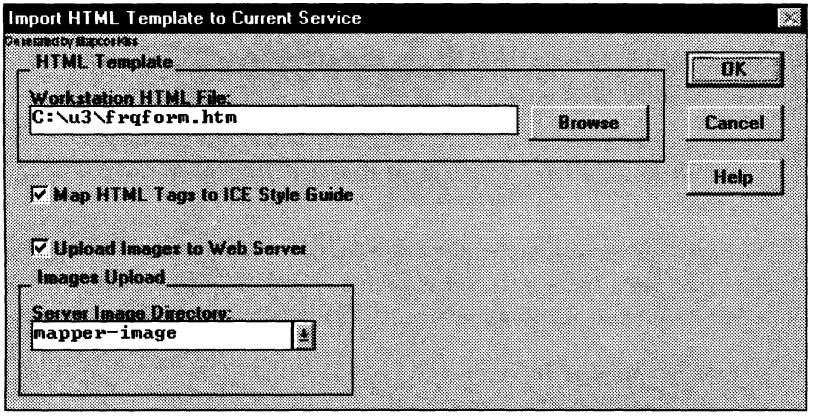
The following describes what happens during the export process:

- Everything within the service between the start and end HTML tags ([HTML] and [/HTML]) will be exported.

- Any MAPPER scripting options will be included and identified using the HTML script tag [SCRIPT] and associated end tag [/SCRIPT]. This means the position of script options relative to HTML tags will be maintained when updating the HTML template using the authoring tool. You can, if required, edit the script itself using the authoring tool; all of the HTML template including the script will be imported back into the service using the Import HTML option.
- References to the Style Guide within the HTML template will be replaced with their respective Style Guide values for the current service.

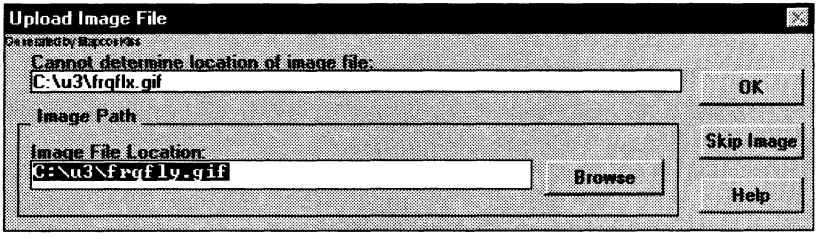
To import an HTML file into an existing service

1. Open the service using the *Open Service* option.
2. Click the Import HTML button on the Service Attributes dialog. This displays the Import HTML Template from Current Service dialog.



3. In the Workstation HTML File, specify the path to the HTML file.
4. If you wish to use the Style Guide information for this service, check the Map HTML Tags to ICE Style Guide checkbox. See "Using a Style Guide" on page 2–13.
5. If the template contains images, check the Upload Images to Web Server checkbox. This ensures that these images are stored in the directory specified in the Server Image Directory selection list. Cool ICE tries to determine the location of an image based on its reference specified in its HTML <IMG> tag. If an image

cannot be found (because it is not accessible from your workstation), you will be asked to specify the exact location or to skip the upload of the image.



Before an image is uploaded to the server, Cool ICE will check if the image file already exists on the server and if so you will be asked to confirm the replacement of the image.

The following describes what happens during the import process:

- HTML tags are mapped to the Cool ICE Style Guide. The HTML template is scanned and selected HTML tags and attributes are translated into the equivalent Style Guide keywords.
- Images are uploaded to the Web Server. The HTML template is scanned for references to images, which are then transferred to the image directory on the Web Server.
- Long lines are automatically wrapped. Lines longer than 80 character will be wrapped at a space or at the end of an HTML tag. When that is not possible, a line will be broken at column 80 and continued on the next line.
- Essential script is inserted. The MAPPER run statements required to turn the HTML template into a dynamic service and to interface to Cool ICE are inserted at the top and the bottom of the template.
- HTML tag delimiters are converted. HTML tag delimiters (< and >) are converted to opening and closing square brackets. This is to avoid confusion with the standard MAPPER variable name delimiters (also < and >). These delimiters are automatically converted back by the Cool ICE Service Handler before a document is sent to the browser.

Other File Options

Details on how to use the other File options (to open, close, save, copy, delete and rename a service) are described in the Cool ICE on-line help.

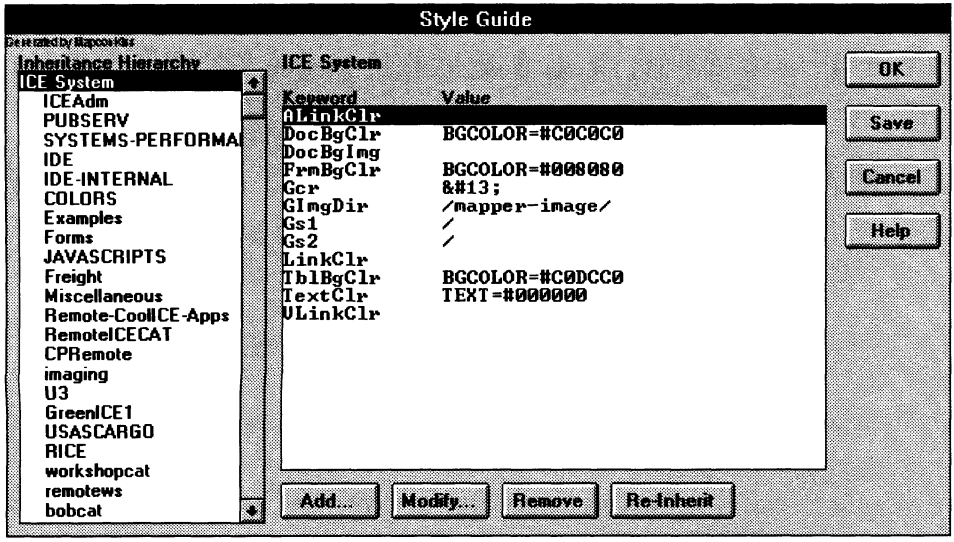
Using a Style Guide

The *Style Guide* option serves two roles, as follows:

- To allow you to provide a consistent look and feel to HTML documents generated by a dynamic service, and to be able to change this at one source.
- To provide an easy way of maintaining common variables to be used in dynamically generated HTML code.

To modify a Style Guide

1. Choose *Options*, *Style Guide*. This displays the Style Guide dialog.



2. From the Inheritance Hierarchy selection list, select the level at which you wish to specify styles.

   The Inheritance Hierarchy list shows the hierarchy of categories and services within Cool ICE. At the top of the category is the Cool ICE system. You can specify a Style Guide for the whole system, for a category or for a service. A Style Guide specified at the system level will automatically apply to all services in the system. A Style Guide specified for a selected category will only apply to a

service within that category. A Style Guide specified for a single service will only apply to that service.

When you select a level, a list of styles is displayed for that level. These show the keyword-values pairs that constitute a particular style.

- The keyword entry is a unique item entry in the Style Guide. At run time the Cool ICE Service Handler will create a MAPPER variable out of the keyword containing the value specified in the Value field. A service can then use the keyword by referencing the equivalent variable.
- The value entry is the content of the keyword.

3. Create or modify the keyword-value pairs for the Style Guide as required.

   The Re-Inherit button resets the Style Guide with the Style Guide defined for the level above it in the hierarchy.

4. When you have finished creating or modifying the Style Guide, press the Save button. Only then are all the modifications you have made saved.

Viewing Events

The *Event Viewer* option allows you to view events logged by Cool ICE. The following types of event logs are kept:

- Access Log. Logs all requests for services made from a browser. This includes information about who requested the service, at what time and how long it took for the MAPPER system to execute the service.
- Error Log. Logs any service failure due to a syntactical error in the service specifications.
- Trace Log. Records information for services that have the Trace On attribute turned on.

Access Log

All requests for services made from a browser are logged by Cool ICE with information about who requested the service, at what time and how long time it took for the MAPPER system to execute the service. This information can be used for different purposes.

- It can provide invaluable information for marketing purposes as to which services are of interest to the users.

- It can be used as a basis for charging for Internet services.
- It can provide information about when the server is busy.

The access log is maintained on a daily basis. A new log report is allocated each day, and will contain all service requests logged for that day. At the first service request of a new day, Cool ICE will summarize the log data of the previous day and add the data to a summary report.

Note: *This summary report is kept at report 8F within the drawer where Cool ICE is installed. Access to the information in the summary report is not provided by any specific Cool ICE Administration option, but it can be accessed through MAPPER for individual customized analysis.*

The *Access Log* option contains two suboptions that allow you to specify the time period you wish to analyze and how you want to view the information:

- Log Settings
- View Access

Log Settings

The *Log Settings* option allows you to configure the log cycle as follows:

- **Daily Cycle**. The Log report will be overwritten every day.
- **Weekly Cycle**. Log reports will be kept on a weekly basis.
- **Monthly Cycle**. Log reports will be kept on a monthly basis.
- **Logging Off**. Logging is turned off and service requests will not be recorded.

View Access

This displays the View Access Log dialog, which provides a summary of the number of hits (service requests) per category and service.

The following standard views are provided:

- **Most Popular Service.** This provides a summary of the number of hits (service requests) per category and service.
- **Average Service Times.** This provides an average service time per category and service. Service time is the time it took to perform the service; the elapsed time measured from when the Cool ICE Service Handler received the request until it sends the HTML document back to the Web Server.

- **User Sign-On.** This provides a summary of the number of times a user ID has been signed on to the system.
- **Services by User.** This provides a summary of the number of hits (service requests) per category and service listed by user ID's.
- **Users by Service.** This provides a summary of the number of hits (service requests) per user ID listed by category and service.
- **Peak Time Hit.** This lists the total number of hits (service requests) for the selected time interval. The default time interval is 60 minutes. You can also select 30 and 15 minute intervals.

View Error Log

Whenever a service fails because of a syntactical error in the service specifications, the error will be logged by Cool ICE. The error log provides the service developer with a lot of information about the service at the time it failed (for example, the date and time of the failure, who requested the service and internal values from the service).

The *View Error Log* option provides a list of entries in the error log. Selecting a log entry from the list will show the information logged for that incident.

View Trace Log

The trace log is a very powerful debugging aid for the service developer who needs to trace/debug a dynamic service in order to analyze a problem. Services requested through the browser are executed in background mode and only output in HTML format can be viewed by the browser. The service developer therefore normally has no means of tracing/debugging a service in the browser environment.

The trace log enables the service developer to run a service in foreground utilizing all standard debugging facilities provided by MAPPER, such as the Run Debugger and use of simple Display functions.

The trace log is associated with the trace On feature within the Service Attributes window. When turned on, the ICE system will log all the input for a service each time it is requested via the browser. When the service request and all the input is captured, we can then run the service in foreground simulating the input is coming from a browser.

The *View Trace Log* option provides a list of entries in the trace log. Selecting a log entry from the list will show the information logged for the incident. To trace

problems in a dynamic service you need to set the *Trace On* option in the Service Attribute dialog for that service. When selected, the service request from the browser including all input to the service, is saved in a trace log.

Security

Cool ICE provides security at run time when services are requested from browsers. At run time, the security profile of the user is matched with the security profile of the service being requested. If the profiles match, the user will be allowed to perform the service.

Security profiles can be specified at the system, category or service levels. Profiles specified at a certain level will automatically be inherited by services below that level. Security profiles at the system level will apply to all services in the whole system. Security profiles specified for a category will apply to all services within that category. Security profiles specified for a service will only apply to that service.

The following is a list of security maintenance options:

- Profiles
- User Registration
- Services Security

Profiles

This is used for setting up a table of valid profiles to be allocated to users and services. A profile must be specified in the Security Profile table before it can be allocated.

The Allocate Profile button displays the Security Profile Allocation dialog. This allows you to allocate or de-allocate a profile to a range of users.

User Registration

This is used for registering users for whom special security profiles need to be allocated.

Not all users need to be registered to use Cool ICE. Users that are not registered in Cool ICE will be allowed access to all open services. A service is open if it does not have special security profiles allocated.

The Profile Membership button displays the User Security Profile Membership dialog. This allow you to give a user membership of a range of profiles, or to take profile memberships away from a user.

The Profile Report button displays the User Security Profile Report dialog. This provides an overview of the users and the profiles to which they belong. By double clicking on a user in the list, the Profile Membership dialog will be opened allowing you to give that user membership to a range of profiles.

Services Security

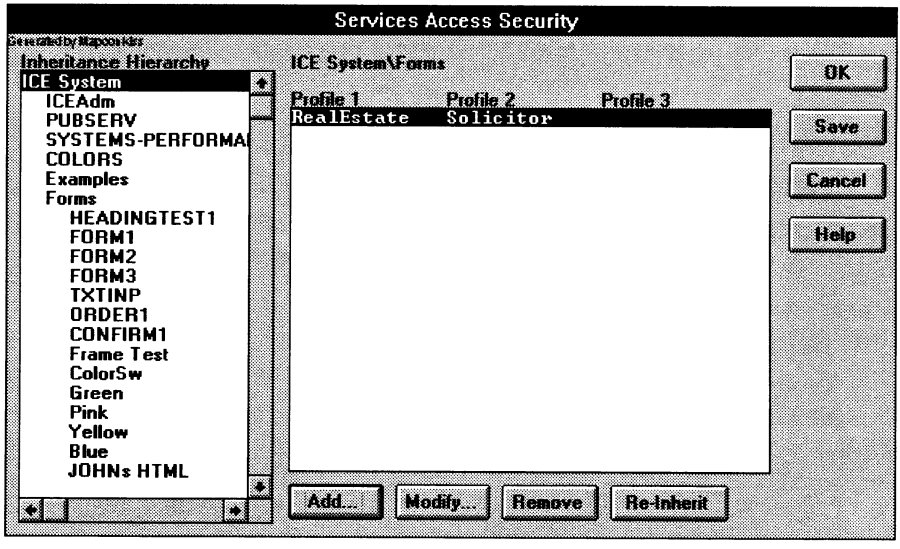
This is used for allocating security profiles for selected services for which special security is required.

Not all services need to be allocated profiles in order to be accessible by users. All users will be given access to all open services.

Services that have been allocated profiles will only be accessible by users with a matching profile.

To allocate a profile

1. Choose *Security*, *Services Security*. This displays the Services Access Security dialog.



The Inheritance Hierarchy column is used for selecting the level at which you wish to specify security. Security can be specified at different levels: a specific

service, a category or the whole system. Security specified at the system level will automatically apply to all services in the whole system. Security specified for a selected category will only apply to services within that category. Security specified for a single service will only apply to that service.

The columns Profile 1, Profile 2, Profile 3 show allocated profiles.

2. Select a level and press the Add button to add a profile to it. This displays the Add Services Access Security dialog.
3. Press OK to update the Services Access Security dialog with your choice of profiles.

You can specify how the profiles are to be matched against the user profiles using logical AND/OR conditions, as shown in the following examples:

- Selecting and inserting 'RealEstate' from Profile 1 and also selecting and inserting 'Solicitor' from the same Profile 1, means that this service can be accessed only by users who are 'RealEstate' ***or*** by users who are 'Solicitor'.
- Selecting and inserting 'RealEstate' from Profile 1 and 'Solicitor' from Profile 2, means that this service can be accessed only by users who are 'RealEstate' ***and*** 'Solicitor'.

Other Administration Options

Document Directory Aliases

This allows you to maintain a table of mappings to directories on the Web Server allocated for storing static HTML document. When defining a static document, this table provides a list of valid document directories to which static HTML documents can be transferred.

Image Directory Aliases

This allows you to maintain a table of mappings to directories on the Web Server allocated for storing image files. This is used when you are importing an HTML template and need to upload images to the Web Server. This table provides a list of valid image directories for uploading images into.

SignOn Settings

This allows you to configure how users coming in to Cool ICE from a browser will identify themselves. When the Cool ICE Gateway has been configured to request user sign-on, the actual sign-on form will be requested from Cool ICE. This dialog window allows you to configure the content of the sign-on form.

Appendix A
The Structure of a Service

A service can be divided into a header section, a footer section and a service body. These sections are described below and shown in Figure A–1 on the following page.

- **Service Header**. Header script is automatically added to a service. This script is required for the service to interface with the Cool ICE Service Handler. The service is called by the Service Handler as a subroutine at label @001. Parameters are passed to the service in the service input report.

  This report contains the following information:

  – Global Cool ICE system information.

  – Style Guide information.

  – Browser input.

  The content of the report is generated as a subroutine with the necessary script for initializing variables with the appropriate content. This allows an unlimited number of parameters to be passed to the service.

- **Service Body.** This contains script for carrying out the *business logic* of the service (for example, for database access), and for outputting HTML code.

- **Service Footer.** The footer script closes the output area and returns control of the HTML document to the Cool ICE Service Handler.

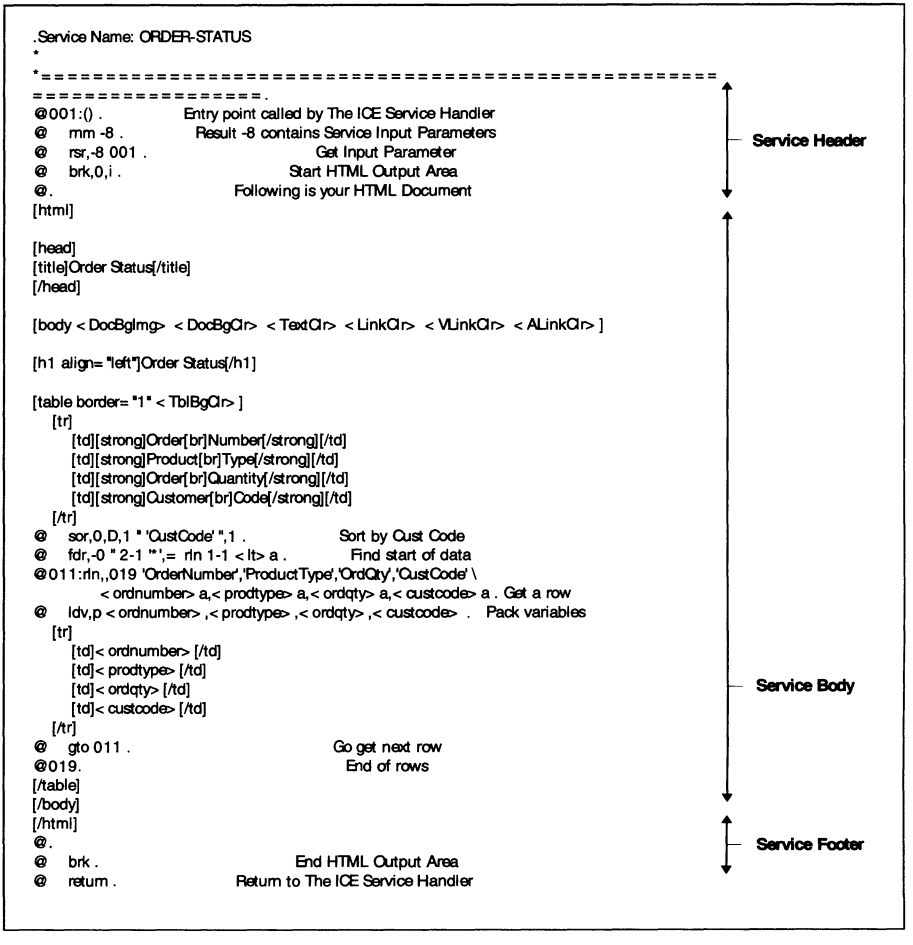


Figure A–1 The Structure of a Service

Service Input Parameters

A service receives input parameters through a result report also referred to as "-8". "-8" contains the necessary script functions for initializing and loading script variables with values of the input parameters. Each input parameter is identified by a name and a value. The name of a parameter becomes the name of a script variable. Before the Cool ICE Service Handler parses control to the service it generates the content of "-8" as a script subroutine and a "load variable" script function for every parameter. "-8" is

a mechanism for passing and receiving an unlimited and also unknown number of parameters. By issuing a "Run Subroutine" (@rsr) script function to "-8", the service receives all input parameters. This is usually done at the beginning of the service as follows:

```
@001:() .          Entry point called by the ICE Service Handler
@   rnm -8 .       Result -8 contains Service Input Parameters
@   rsr,-8 001 .   Get Input Parameters
```

"-8" consists of the following sections:

- **Multi-Line Fields**. This provides a mechanism for receiving input from HTML forms including objects like multi-line text areas and multi-selection list boxes. This section is only present when an HTML form includes any of these objects.
- **System Variables**. This section include parameters containing information and values from the Cool ICE environment. This will have various uses depending on the service.
- **Style Guide**. This contains all keywords and values as defined in the style for a particular service.
- **Browser Input**. This section contains parameters and values received from the browser. It contain parameters originating from the HTML hyperlink tag (HREF) and the action field of forms. Form input objects and associated value/content are also present here. An HTML form input object can be referenced as a script variable by the equivalent name as specified in the HTML form.

The following is an example of "-8":

```
@001.
@. **** BEGIN MULTI-LINE-FIELDS ****
@IF <MULTILINE> = 'M-Positive',(101) .
@. **** END MULTI-LINE-FIELDS ****
@. **** BEGIN SYSTEM VARIABLES ****
@LDV,p <GPath>s40='c:\netscape\server\mapimg\' .   Graph Directory
@LDV,p <GDrilFile>s50='C:\temp\G00864' .           Driller Input File
@LDV,p <GDrilErr>s50='DRLERROR.GIF' .              Driller Error Image
@LDV,p <GDrvCab>a4=340 .                           Driver Cabinet Location
@LDV,p <GDrvDrw>a1=B .                             Driver Drawer Location
@LDV,p <GCurSvc>s20='POST-QUERY' .
@LDV,p <SvcTitle>s80='Post Query - forms input received.' .
@LDV,p <GPathInfo>s255='/SessionId=855332521MPR86/Examples/POST-
QUERY
```

```
@LDV,p <GUrl>s255='http://192.39.97.61/ICEGate/SessionId=855332521MPR86'
.
@LDV,p <GSysImgDir>s40='/mapper-image/' .        Dir Alias of Images
@LDV,p <GSysKwd1>a10=Category .                  Category Keyword
@LDV,p <GSysKwd2>a10=Service .                   Service Keyword
@. **** END SYSTEM VARIABLES ****
@. **** BEGIN STYLE GUIDE ****
@LDV,p <ALinkClr>a1='' .
@LDV,p <DocBgClr>a15='BGCOLOR=#C0C0C0' .
@LDV,p <DocBgImg>a1='' .
@LDV,p <LinkClr>a1='' .
@LDV,p <TblBgClr>a15='BGCOLOR=#C0DCC0' .
@LDV,p <TextClr>a12='TEXT=#000000' .
@LDV,p <VLinkClr>a1='' .
@. **** END STYLE GUIDE ****
@. **** BEGIN BROWSER INPUT ****
@LDV,p <Category>s20='Examples' .
@LDV,p <Service>s10='POST-QUERY' .
@LDV,p <TRANSID>s11='00234-00864' .
@LDV,p <SESSIONID>s14='855332521MPR86' .
@. **** END BROWSER INPUT ****
@ESR .
@101. **** MULTI-LINE-TEXT - M-Positive ****
Cool ICE
is
very good
@ESR .
```

Specifying Hyperlinks

Hyperlinks in an HTML document are used to call a service and pass parameters to it. Hyperlinks are defined by the following HTML anchor tag:

<A HREF="...></A>

A hyperlink to a service must specify the full URL (see the section, "URLs" on page 1–5) to the service, as follows:

http://machine-name/gateway/category/service/keyword=value/...

In order to make a service independent of the machine name and gateway, a global variable <GURL> is defined by the ICE Service Handler.

The following example illustrates the use of the <GURL> variable. It shows a menu of four options, which provide hyperlinks to different services:

General Document Enquiry
Order from the Publication Services
Check Order Status
New This Week

The underlying code for this page is as follows:

```
. Service Name: Main-Menu
*
*=============================================
@001:() .
@    mm -8 rsr,-8 001 .
@    brk,0,i .
[HTML]
[HEAD]
[TITLE] Publications Services [/TITLE]
[/HEAD]
[BODY]
[H1][CENTER] Publication Services [/CENTER][/H1]

[B]Select the required service.[/B][BR]

[A HREF= "< GURL> /Pubserv/ENQ1 "]General Document Enquiry[/A][BR]
[A HREF= "< GURL> /Pubserv/ORD1 "]Order from the Publication Services[/A][BR]
[A HREF= "< GURL> /Pubserv/OST1 "]Check Order Status[/A][BR]
[A HREF= "< GURL> /Pubserv/NTW1 "]New This Week[/A][BR]

[/BODY]
[/HTML]
@    brk .
@    return .
```

URLs for services, using Cool ICE <GURL> variable.

Category and service names can be specified either as positional:

<GURL>/Pubserv/ENQ1

or as keywords:

<GURL>/Category=Pubserv/Service=ENQ1

Keywords can be specified in any order.

Hyperlink Parameters

A hyperlink can also be used to pass parameters to the service being called. Parameters are specified as pairs of 'keyword=value', separated by a slash. The receiving service will receive the parameters as variable names containing the value.

In the following example, the service CustInfo will receive a variable <CustNo> containing the value AMCO:

```
[A HREF="<GURL>/Forms/CustInfo/CustNo=AMCO"]AMCO Customer Information[/a]
```

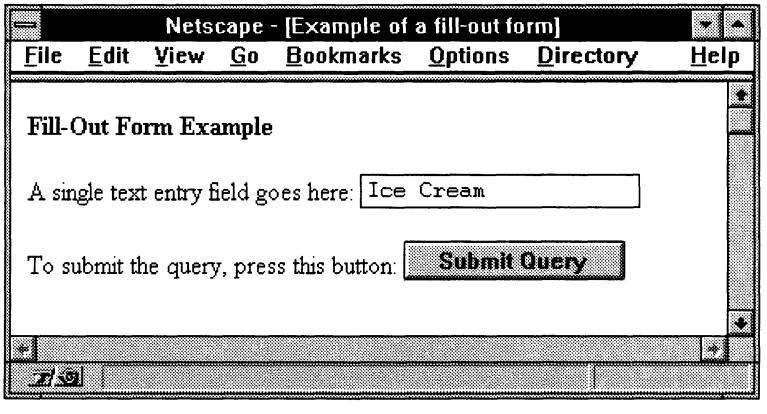
Spaces in a Hyperlink Reference

Web browsers require that hyperlink parameters do not contain spaces. The character "%20" is therefore used as a substitute for a space. Following is an example of script for converting spaces to "%20" when generating hyperlink parameters:

```
@ ldv,p <param>s80='CustName=<custname>' .
@ lcv t~b99 <param> ' ' ,<voccs>i6 ldv <href>s80=<param> .
@ if <voccs> le 0 . ; lcv t~b<voccs> <href> ' '/%20 .
@ ldv,p <href> .
[A HREF="<GURL>/Forms/CustInfo/<href>"]Customer Information[/A]
```

Handling Forms

The following is an example of how to send a form input entry to a service:



The HTML service for this form includes the following code:

```
[FORM METHOD="POST" ACTION="<GURL>/Examples/POST-QUERY"]
[B] Fill-Out Form Example [/B][P]
A single text entry field goes here: [INPUT NAME="entry"] [P]
To submit the query, press this button:
   [INPUT TYPE="submit" VALUE="Submit Query"] [P]
[/FORM]
```

The first line in the above code includes the form action field, as follows:

ACTION="<GURL>/Examples/POST-QUERY"]

This specifies the service to be called when the user clicks the Submit Query button. See "Specifying Hyperlinks" on page A–4.

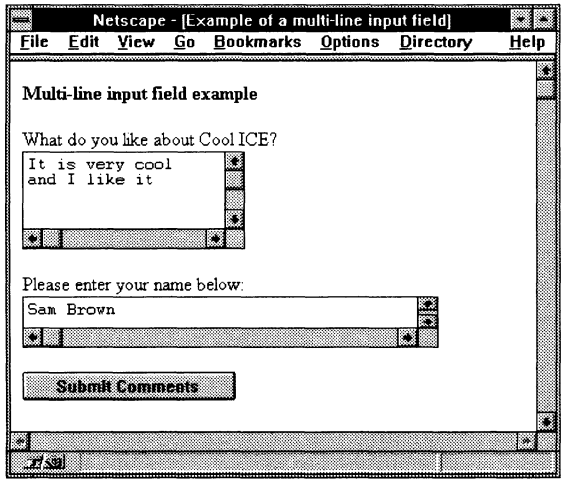
The third line in the code above includes the form entry fields, as follows:

[INPUT NAME="entry"]

This is passed to the service specified in the form action field. The receiving service refers to the input field names as script variable names. In the example above, the field name "entry" is referred to as <entry>, and will contain the word "Ice Cream".

Handling Multi-line Objects

Multi-line objects are form text-areas and multi-selection list boxes. The following is an example of creating a multi-line input field for a service:



The following is the code for this form:

```
[FORM METHOD="POST" ACTION="<GURL>/Examples/POST-QUERY"]
[B]Comments And Feedback Welcome[/B][P]
What do you like about Cool ICE? [BR]
[TEXTAREA NAME="M-Feedback" ROWS=4 COLS=20][/TEXTAREA] [P]
Please enter your name below: [BR]
[TEXTAREA NAME="M-Username" ROWS=1 COLS=40]Your Name
Here[/TEXTAREA] [P]
[INPUT TYPE="submit" VALUE="Submit Comments"] [P]
[/FORM]
```

The form's input parameters are received by the service as shown below (note the multi-line entries for M-Feedback and M-Username).

```
@LDV,p <Category>s20='Examples'
@LDV,p <Service>s10='POST-QUERY'
@LDV,p <TRANSID>s11='00466-00215'
@LDV,p <SESSIONID>s14='850797255MPR20'
@101.****MULTI-LINE-TEXT - M-Feedback****
It is very cool
and I like it
@ESR
@102****MULTI-LINE-TEXT-M-Username****
Sam Brown
@ESR
```

The name of a multi-line object must be prefixed by an "M-" in order to distinguish it from a single-line object.

A service requests input from a multi-line object by specifying the name of the object in a script variable "<MULTILINE>", and calls the input parameters report "-8". The content of multi-line objects are presented to the service as a script result report, as follows:

```
@ brk,0,a .
@ ldv,p <MULTILINE>a12=M-COMMENTS .
@ rsr,-8 001 .
@ brk
```

Glossary

Category

Services are grouped into categories. A category may group a set of services that form an application or some other relationship; it is up to the developer as to how they wish to categorize services. A category may be a *local category* (the services for the category are stored on the local server) or a *remote category* (the services for the category are stored on a remote server).

Dynamic Service

A dynamic service generates content and HTML code at the time of invocation. It may access a local or commodity database, manage inquiry forms and so on. Dynamic services can be developed by adding script to an HTML page, or they can be built from scratch. They can include complex application logic, access to multiple databases, graphic art, dynamic business graphs and static text. A dynamic service can be triggered as result of an end-user action or as a result of a scheduled request.

Gateway

The Common Gateway Interface is a means for a Web server to talk to programs on another machine. The idea is that each server and client program, regardless of the operating system platform, adheres to the same standard mechanism for the flow of data between client, server, and gateway program.

Hyperlink

A hyperlink is a connection from a Web page to a resource on the Web. The destination of the hyperlink can be another page, a multimedia file, a program or a service. Hyperlinks are usually short pieces of text and a browser emphasizes hyperlinks by underlying and displaying them in a specific color. When a user clicks on a hyperlink, the browser passes the request to the appropriate Web server and waits for a page to be returned. The destination of the hyperlink is encoded as a URL.

Service

This is the name used by Cool ICE for any service provided by a company via the Internet or a corporate Intranet. Cool ICE distinguishes between static and dynamic services.

Statelessness

After the server responds to a client's request, the connection between client and server is dropped. Statelessness also means, from the standpoint of the Web developer, that there is no memory between client connections; there is no trace of recent activity from a given client address, and the server treats every request as if it were brand new — that is, without context.

Static Service

A static service allows a user to access documents that do not include added script. These are called static documents. They usually consist of HTML code and can be developed using any standard Web page development tool. Once created, they may be registered for access to designated users.

URL

Uniform Resource Locator. This is the basis for referring to resources (files, services and so on) on the Web. A URL consists of a string of characters that uniquely identifies the resource. When a Web browser opens a particular URL, the user gains access to the resource referred to by the URL.

Index

N

P

R

S

T

U

V

W

Tape

Do Not Staple

Please Fold and Fasten

Tape

Fold here

Airmail

Correct Airmail
Postage Required

Unisys Australia Pty Ltd
115-117 Wicks Rd
North Ryde, NSW 2113
Australia

Attention: ACUS Product Information

UNISYS

Help Us to Help You

Unisys Corporation is interested in your comments and suggestions regarding this document. We will use them to improve the quality of your product information. Please check the type of suggestion:

○ Addition ○ Deletion ○ Revision ○ Error

Publication title

Document number Date

Comments:

Name Telephone number ( )

Title Company

Address

City State ZIP code

78460847-000

May 1997

Dear Mr Weston

I am pleased to inform you that Unisys has announced the first release of the Cool ICE for WINDOWS NT, a new product that provides leading edge technology as an Internet Commerce Enabler. Cool ICE allows the creation of new web applications using existing applications and data, which may be in a variety of different databases on various platforms, effectively turning them into electronic commerce solutions. The underlying application continues to run as is, presented to off-web users in the same manner to which they are accustomed. But as presented on the website, its workflow and logic may be manipulated by the enterprise to cater to multiple user groups. The legacy data is treated as a resource.

Cool ICE includes a rapid application development environment based on a scripting language for linking different databases with the Cool ICE repository. Sources of the data can include Oracle, Sybase, Informix, IBM DB/2, DMSII, RDMS, LINC and MAPPER databases via a native database interface. Still other databases can be accessed via Microsoft Open Database Connectivity (ODBC) software.

HIGHLIGHTS:

Cool ICE is a software integration solution that allows organisations to manage a mixture of static and dynamic Internet Web services on a corporate Intranet or the public Internet.

The Cool ICE solution for Windows NT provides:

- Management of Internet documents and dynamic Internet services
- Building dynamic Internet business services based on existing applications and data
- Secure Web access to applications on existing servers.
- User based views of data and services

COOL ICE BENEFITS:

The benefits of using Cool ICE include:

- A minimal risk implementation as the process does not alter existing application code or current IT infrastructures.
- Powerful information Web site management provided by the Cool ICE "Coolware".
- Improved customer service and reduced costs as customers/partners/employees have direct, instant access to products and services on the World Wide Web.
- Extends the reach of existing products and services into new markets.
- Enterprise class attributes: scalability, resiliency, security

I have included a copy of the Cool ICE User Guide so you can see for yourself how much easier it will be to create Web based applications using this technology.

If you have any questions or would like further information on Cool ICE please contact Unisys Partnership Marketing 1800 020 640.

Your sincerely

V Herring

Vic Herring
Solutions Marketing Manager
Unisys Australia