Oracle® Scripting

Implementation Guide

Release 11*i*

August 2000

Part No. A86115-01

ORACLE®

Oracle Scripting Implementation Guide, Release 11*i*

Part No. A86115-01

The Programs are not intended for use in any nuclear, aviation, mass transit, medical, or other inherently dangerous applications. It shall be the licensee's responsibility to take all appropriate fail-safe, backup, redundancy, and other measures to ensure the safe use of such applications if the Programs are used for such purposes, and Oracle Corporation disclaims liability for any damages caused by such use of the Programs.

Contents

Oracle Scripting Implementation Guide

Send Us Your Comments

Oracle Scripting Implementation Guide, Release 11*i*

Part No. A86115-01

Oracle Corporation welcomes your comments and suggestions on the quality and usefulness of this document. Your input is an important part of the information used for revision.

- Did you find any errors?
- Is the information clearly presented?
- Do you need more information? If so, where?
- Are the examples correct? Do you need more examples?
- What features did you like most?

If you find any errors or have any other suggestions for improvement, please indicate the document title and part number, and the chapter, section, and page number (if available). You can send comments to us via the postal service.

Oracle Corporation
CRM Content Development Manager
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

If you would like a reply, please give your name, address, telephone number, and (optionally) electronic mail address.

If you have problems with the software, please contact your local Oracle Support Services.

Preface

Welcome to the Oracle Customer Relationship Management, Release 11*i*, suite of applications.

This Implementation Guide provides general descriptions of the setup and configuration tasks required to implement Oracle Scripting successfully.

This preface explains how this Implementation Guide is organized and introduces other sources of information that can help you.

Intended Audience

This guide is aimed at anyone who is tasked with implementing Oracle Scripting, including:

- Database Administrators
- System Administrators
- Technical Specialists

This guide assumes you have the following prerequisites:

1. Understanding of computer-telephony integration (CTI)
2. Understanding of call center technology
3. Understanding of the company business processes
4. Understanding of Oracle Applications, Release 11i
5. Understanding of Oracle Workflow

Structure

This manual contains the following sections:

- Introduction
- What is Scripting?
- Considerations for Planning an Oracle Scripting Implementation
- Typical Release Dependencies
- Related Documentation and Resources
- Setting Up Oracle Scripting
- Creating a Script
- Compiling and Deploying a Script
- Launching a Script
- Setting Profile Options
- System Profile Options
- Configuring and Testing Integration Points
- Workflows
- Converting and Inputting Existing Data
- Testing an Implementation Project
- Considerations for Future Upgrade Paths
- Acronym Glossary

Related Documents

For more information, see the following manuals:

- *Oracle Applications Concepts*
- *Oracle Applications Product Update Notes, Release 11i*
- *Oracle Applications Release Notes, Release 11i*
- *Oracle Applications Installation Update Notes*
- *Installing Oracle Applications, Release 11i*
- *Maintaining Oracle Applications, Release 11i*

- *Upgrading Oracle Applications, Release 11i*
- *Oracle Applications System Administrator's Guide*
- *Oracle Applications User's Guide*
- *Oracle Workflow Guide*
- *Implementing CRM Applications*
- *Oracle Scripting Concepts and Procedures*
- *Oracle Scripting Technical Reference Manual*

Oracle Scripting Implementation Guide

Introduction

Oracle Scripting is a script development tool available to Oracle Customer Relationship Management (CRM) 11i users. Unlike many of the CRM applications, such as Oracle TeleSales or Customer Support, a script is always customized for each new Oracle customer. Each customer has a unique set of requirements for what they need in a script, and part of the Oracle requirements gathering process is to work with the customer to understand the requirements and develop a script that meets the enterprise's needs. This means that the implementation process--which consists of confirming that the product has been installed correctly and then making the customized script(s) available, is somewhat different from that of other CRM applications.

The Oracle Scripting Engine, included in the CRM "Rapid Install", makes the Scripting runtime environment available to those CRM 11i Applications that take advantage of its capabilities. Additionally, scripts can be developed in Oracle Scripting Author, provided as part of the Oracle Call Center Applications CD (separately installed).

This guide is intended for use by those implementing Scripting Author to develop or modify scripts and execute them through CRM 11i or custom applications. Oracle Scripting is described below in further detail, followed by the following topics:

- Considerations for Planning an Oracle Scripting Implementation
- Typical Release Dependencies
- Related Documentation and Resources
- Setting Up Oracle Scripting Author
- Configuring and Testing Integration Points

- Workflows
- Converting and Inputting Existing Data
- Testing an Implementation Project
- Considerations for Future Upgrade Paths
- Acronym Glossary

What is Scripting?

Scripting is a generic term for the concept of providing customer service representatives or “agents” a method of guided interaction with a customer or prospect. Scripting applications automate the process by incorporating business rules into a software application. Typically, a script consists of a series of presentation panels, or questions, which guide an agent through a conversation with a customer. A script may be as simple as a set of survey questions, or it can be a fairly complex set of questions and answers that guide an agent among various screens within an application.

Oracle Scripting is designed to meet the needs of call centers across a broad range of customer information needs. Call centers are increasingly using other methods to satisfy customers (email, the Internet, etc.), and are now often referred to as "customer interaction centers." (This document uses both terms interchangeably.)

The primary goal of a script is to provide customer interaction center agents with the tools they need to easily respond to or present information to customers in a consistent manner. Through defining effective scripts, customer interaction centers can reduce the time it takes to train agents, make the agents more effective and productive while they are on the telephone with customers, and insure that all agents have the same source of information at their fingertips.

Because Oracle Scripting is targeted at a very specific subset of customer interaction center requirements, Oracle Consulting should install Oracle Scripting as a *part* of another, broader set of CRM applications. Typically, a script will be designed to integrate with an application such as Oracle Customer Support (a component of Oracle TeleService, and formerly known as Oracle Customer Care) or Oracle TeleSales. The CRM application performs most of the "contact management" and provides all essential data collection and reporting; the Oracle Scripting component gives the agent additional information that makes it easier to establish a dialog with the customer by insuring that the agent has a clearly guided presentation.

What Are the Components of Oracle Scripting?

Oracle Scripting consists of two components:

- Script Engine, which provides an easy-to-use GUI at the agent desktop
- Script Author, which provides a visual layout so interaction center staff can lay scripts out graphically

Thus, Scripting provides both a development environment for interaction centers, and a runtime environment in which agents can be provided a method of being guided through their conversations with customers. As the customer interacts with an agent (whether by providing yes-or-no responses, or by inquiring about products and services), the agent moves to appropriate text screens providing the information needed to respond fully and accurately.

Scripts can be created with multiple skill levels in mind, enabling a call center to provide detailed scripts to novice agents, and more flexible routing throughout the script to more experienced agents. (In future releases, pre-defined views and information will be presented based on the agent’s responsibility code.) Either method enables a script author to create a single script that satisfies several agent groups. The script can be created in a timely manner and deployed to all agents - at the same time!

Oracle Scripting is currently integrated with one application within the Oracle eBusiness Suite and, with the release of 11.5.2, will integrate with several more in the CRM application space. For more information on Scripting integration, see *Scripting and Integration of Other CRM Applications* in this document.

Oracle Scripting is a thin-client application implemented in a multi-tiered architecture on Oracle *8i*. It provides a unified desktop for multiple external systems because it can obtain data from those systems and display the data in a logical, consistent GUI. In most call centers, agents are required to deal with up to twenty different applications when interacting with customers. Oracle Scripting can guide the agent’s workflow among these systems by pre-positioning the most important customer data where it is easy to find. This can result in significantly reduced training time, increased agent productivity, enhanced service and reduced cost of call center operations.

Considerations for Planning an Oracle Scripting Implementation

Purpose for Oracle Scripting Implementation Planning

The importance of implementation planning for Oracle Scripting cannot be overemphasized. The primary purpose for this planning is the successful implementation of Scripting. As noted in the Overview section of this document, Oracle Scripting differs from other CRM applications because it is not an application, but a development tool designed to create call guides for customer interaction centers. Thus, the process of implementing scripting is actually a part of the customization process that Consulting must undergo as part of any new CRM implementation.

Implementing scripting is a complex process that requires knowledge of a variety of technologies and processes. Without the appropriate planning it is easy to result in a mismatch of requirements, underestimate the appropriate hardware, software, or network configuration, or end up with substandard system performance.

Recommendations for Planning Scripting Implementations

Discovery Process

The actual scope of a Scripting project can be deceptive in nature. While at first it may seem that the scoping may include just the business logic of the script and the particulars of the scripting implementation, a closer examination reveals that there are three components: the text (the question / answer portion of the script provided by the client for the call guide), the logic (hopefully worked out in detailed flow charts to ensure call and data flow), and the technology layer.

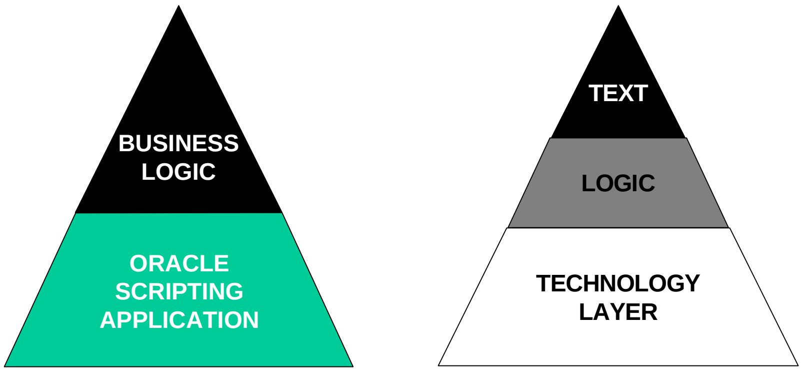


Figure 1. Early views of the Scripting scoping and discovery process

It is the technology layer that can require the bulk of time on a project, although confirming the first two portions (and implementing strict change control on the client once the scope has been agreed to) are key elements of success.

Additionally, business analysis will be required throughout the project, often interrupting development progress. This is because a more detailed understanding of the system being constructed -- only available mid-stream -- is likely to result in adjustments to the development approach, scheduling, and project resources. An experienced project manager will recognize these realities and plan for adjustment, rather than struggle with carrying out a plan conceived before a thorough understanding of the complexities of a scripting implementation were obtained.

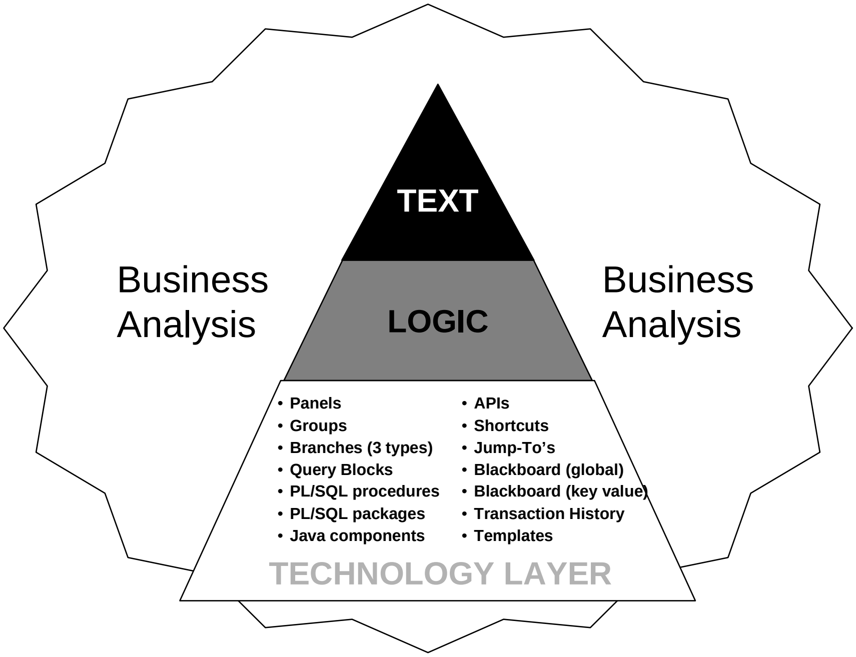


Figure 2. An informed view of Scripting scoping and discovery considerations

The analysis of the technology layer reveals many technical aspects to be considered; it may be hard to scope them, and if the project allows, a multiple-phase process is recommend, which allows for benchmarking which can be used to determine average completion time for the appropriate tasks.

Integration Benchmarking

Oracle Scripting can be integrated with other Oracle CRM 11i applications in one of three ways:

- You can display data from another application within the context of the script;
- You can launch external applications from the script; or
- You can launch the script from other applications.

Ensure that you understand the nature of the desired integration. This includes a determination of whether the application exposes APIs that can be accessed by Scripting, and that is compatible with the technology stack.

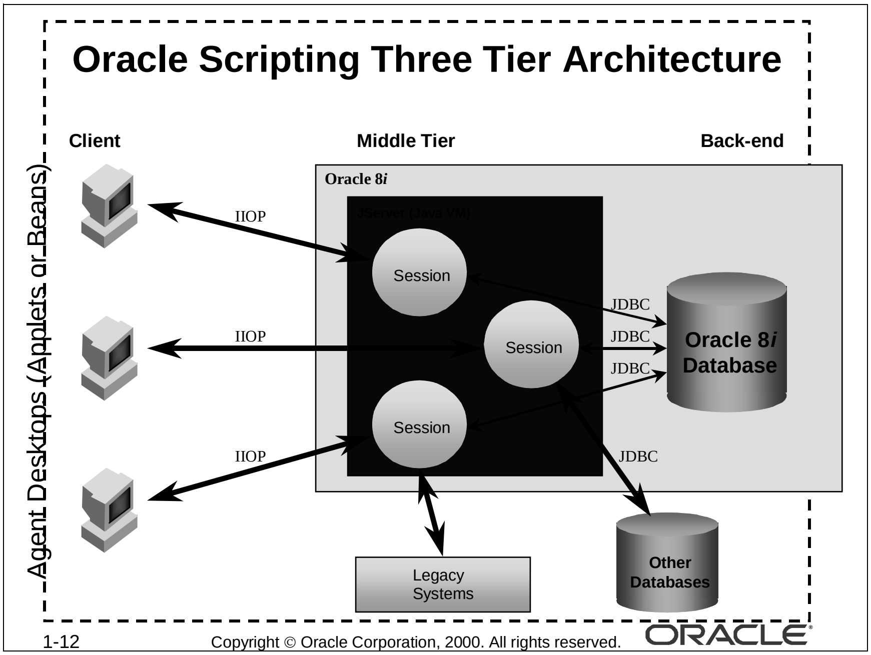
If substantial integration requirements are indicated, it may also be in the best interests of the project to benchmark a limited integration to determine the true level of effort before scoping a larger integration effort. This will help in scheduling and resourcing.

Application Architecture

Oracle Scripting is a pure Java application which uses Oracle *8i* JServer to serve as the middle tier. Because Oracle *8i* JServer acts as the application server, the Oracle Application Server is *not* required. Instead, all of the necessary components of Oracle Scripting are installed as part of the Oracle CRM "Rapid Install" followed by installation of the Script Author via the CRM Call Center Applications ARU (which replaces the CRM Call Center Applications CD - see Step 7 of *Installation steps* in this document).

As an implementation consideration, you need to be aware that the Scripting Engine is installed as a standard 3-tier application. The Scripting Author is installed on a client (Windows 95 or NT) in a 2-tier architecture. However, the Scripting

Author deploys the customized scripts to the *8i* database.



Oracle Scripting Three Tier Architecture

Client

Middle Tier

Back-end

Agent Desktops (Applets or Beans)

1-12 Copyright © Oracle Corporation, 2000. All rights reserved. ORACLE

Figure 3. The three-tiered architecture of Oracle Scripting

Java Technology

Scripting is a 100% Java application, including the client and middle tiers as well as the Script Author.

Although Scripting is not built with Oracle Forms, scripts can integrate with Forms-based applications. In future releases, several out-of-the-box integrations with CRM applications will exist. For more information on Scripting integration, see *Scripting and Integration of Other CRM Applications* in this document.

Java Technology - Oracle *8i* JServer

JServer is a new technology that first became available with the initial release of Oracle *8i*. The Oracle *8i* JServer hosts and runs Java objects in a special Java Virtual Machine (JVM) built into the database.

Since JServer is built in the database, it takes advantage of mature, critical services that have been tested and improved extensively over the years. Scripting uses the Corba ORB for client-to-middle-tier communication.

Finally, JServer includes a JDBC driver that was developed to make the most of being in the same process as the database and bypasses the OCI and network layers, thereby making it more efficient.

Application Architecture - Three-Tier Architecture

Oracle Scripting 11i uses the JServer to achieve a mult-tiered architecture. The data tier is represented by the CRM *8i* database, as well as other databases and legacy systems that contain data needed by the script. The *8i* JServer represents the middle tier, where the scripting sessions are run. Each session represents an agent logged in to the Scripting server. These sessions communicate with the *8i* database via the special native JDBC driver and with other databases and systems via JDBC or other protocols. Last, we have the agent desktops, which communicate with the Sessions in the JServer via Corba and can take the form of either standalone Java applets or Java Beans embedded in another application on the client machine.

Essentially, execution of a script calls the script from the Oracle Scripting schema in the database, executing the GUI or presentation logic on the client and using the processing power of the server to drive the business logic.

Technical documentation regarding Scripting, its architecture, Application Program Interfaces (API), and other material can be found in two places on the Oracle intranet; see *Related Documentation and Resources* in this document.

Implementation Roles and Functions

There are four basic roles for implementing Oracle Scripting:

- Database Administrators (DBAs)
- Systems Administrators (SysAdmins)
- Script Authors
- Developers

Database Administrators

In order to properly implement Oracle Scripting, a DBA with knowledge of Oracle *8i* Relational Database Management System (RDBMS) is essential. Specific required areas of knowledge include:

- Setting up and implementing Java in the database (the JVM);
- Establishing and administering the IIOP port;
- CORBA, IIOP, and JNDI;
- Using LoadJava and DropJava utilities to move required code (such as JAR files containing Java class files which support specific Oracle Scripting scripts or call guides) into the database; and
- Setting, logging and monitoring Trace Files.

Systems Administrators

Setting up the scripting environment can be complicated, and essential to success of an implementation is the assistance of one or more SysAdmins, who must have knowledge or prior experience in:

- Applications environment preparation and setup;
- Applications mid-tier configurations;
- Configuring JInitiator Client settings; and
- Installing and patching Oracle Applications.

Script Authors

Script authors need to leverage the tools inherent in the Script Author development environment to build scripts that are often large and complex. Strong analytical skills, consistency, and the ability to develop scripts from complex flow charts are essential. Script authors should have a good understanding of solid programming concepts (including naming conventions and principals of design), although they should also have Developers and Systems Architects accessible to consult with regarding script development approaches and techniques. Constant assessment of requirements provided to the development team is also required of the script author. In instances where more than one person is developing a script, excellent communication skills are also required. See the section *Challenges to Scripts Built in Parallel with Multiple Developers* in this document for more details. The script author role is an excellent training ground, exposing staff to a variety of technologies that may soon enable persons in this role to expand her or his knowledge to the point where they can move into the more technical Developer role, as described below.

Developers

Developers are also required for most Scripting development and implementation projects. One main role of developers will be to perform integrations between

Scripting and other applications (particularly Forms-based applications). These developers must be familiar with the Script Author development environment, as they may be called upon to develop complex scripts (or scripting groups within a script) that less experienced script authors may not possess the technical skills to perform. Developers also may be required to embed complex logic in a script by developing custom code (Java, PL/SQL, database integration, etc.). This code would typically be associated with a script using the Commands dialog box in the Script Author. Such complex logic would be required to enforce business rules provided to the script development team, preferably in the form of carefully detailed flow charts. Constant assessment of the requirements provided to the development team is also required of the developer. These resources need to work in close collaboration with the chief script author(s) of the script, requiring strong communication skills in addition to technical demands.

End-User Roles and Functions

There are two basic end-users for Oracle Scripting:

- Call center agents, who use the runtime Scripting GUI to guide them through interactions with customers as they talk on the telephone
- Script authors, who actually develop scripts to meet the call center needs

Call Center Agents

Agents are the people who use the completed, runtime scripts as guides when they interact with their customers. Oracle Scripting presents a very simple GUI that makes it very easy for agents to navigate.

Script Authors

Oracle Consulting will typically deliver at least one "template" or baseline script to the customer as part of the CRM implementation process. Regardless, most customer interaction centers have extremely urgent needs to rapidly create and modify scripts for their production environments.

Whether script authors are Oracle Consulting staff or employed by the licensed users of Scripting at the client site, the following areas of knowledge are strongly recommended for individuals developing scripts:

- Knowledge of solid programming concepts;
- Java development skills;
- PL/SQL and database programming;

- Integration with other Oracle applications and Oracle Forms; and
- Strong analytical skills and the ability to map out business flow for script planning.

Overview of Typical Business Processes for Oracle Scripting

The following business processes are involved in the use of Oracle Scripting:

- Script Creation
- Agent/Application Linkage
- Script Deployment

Script Creation
Link to Application
Script Deployment

Figure 4. Business processes involved in using Oracle Scripting

Script Creation

Script creation refers to the process of defining the text, branching logic, and the technology layer for a script. The text and branching are a graphical representation of the question/answer flow through a *call guide* (a call center-specific term which refers to, in the case of Oracle Scripting, a script developed in Script Author to enable a call center agent to navigate through a carefully-scripted set of interactions with the customer or prospect). The term *technology layer* refers to the additional considerations that must be made in defining a script during implementation. These tasks are associated with all of the logic and data manipulation that must occur in the background, transparent to the user. The development of the technology layer typically includes using one or more of the following types of procedures:

- Identify data needed from external applications - The script often needs to reference demographic, product, and other information needed during the context of the call. For instance, the customer's name and address may be validated as part of the script logic.
- PL/SQL processes and packages and Query blocks - To perform activities such as validating users, accessing data in external tables (when it's not already available via published APIs), including data in non-Oracle CRM applications.

- Java commands - To perform conditional logic, set up constants and other variables, and to manipulate both data and call flow as part of the customer/agent interaction.
- Blackboard commands - Scripting uses the blackboard, an area of memory within the Scripting Engine which stores values of embedded values, responses to previously-asked questions, and data passed to the script from external applications.
- Shortcuts - Quick access points, both within the script (for example, go to a wrap-up group) and external to the script (for example, jump to a specific URL on the corporate intranet).

To assist in some of the tasks associated with the technology layer, several template scripts have been defined, called Best Practices scripts. A library of Best Practices scripts will be delivered with the patch to Oracle Scripting 11i R2 (11.5.2), which will define some commonly-used functions, including: customer demographic information, callback information, call wrap-up, and “do not call” templates. These templates can be re-used and/or modified in new scripts, thus reducing the development tasks associated with these types of activities. However, any new script is likely to require additional data beyond that supplied in the Best Practice templates. Refer to the *Oracle Scripting Concepts and Procedures Manual* for more information on Best Practice Scripts and their locations.

Finally, the script must be integrated with the desktop application. Oracle Scripting is designed to act in conjunction with other CRM applications, and we do not recommend its use as a standalone product. The reason for this is that Scripting is not designed to act as a contact management system; it does not collect the detailed statistic data that is so crucial to most interaction center enterprises. It lacks detailed reporting, CTI statistics, data validation, and other benefits that are provided by CRM applications such as Oracle Customer Support (a module of TeleService), or Oracle TeleSales.

The processes associated with creating a script consist of the following high-level activities.

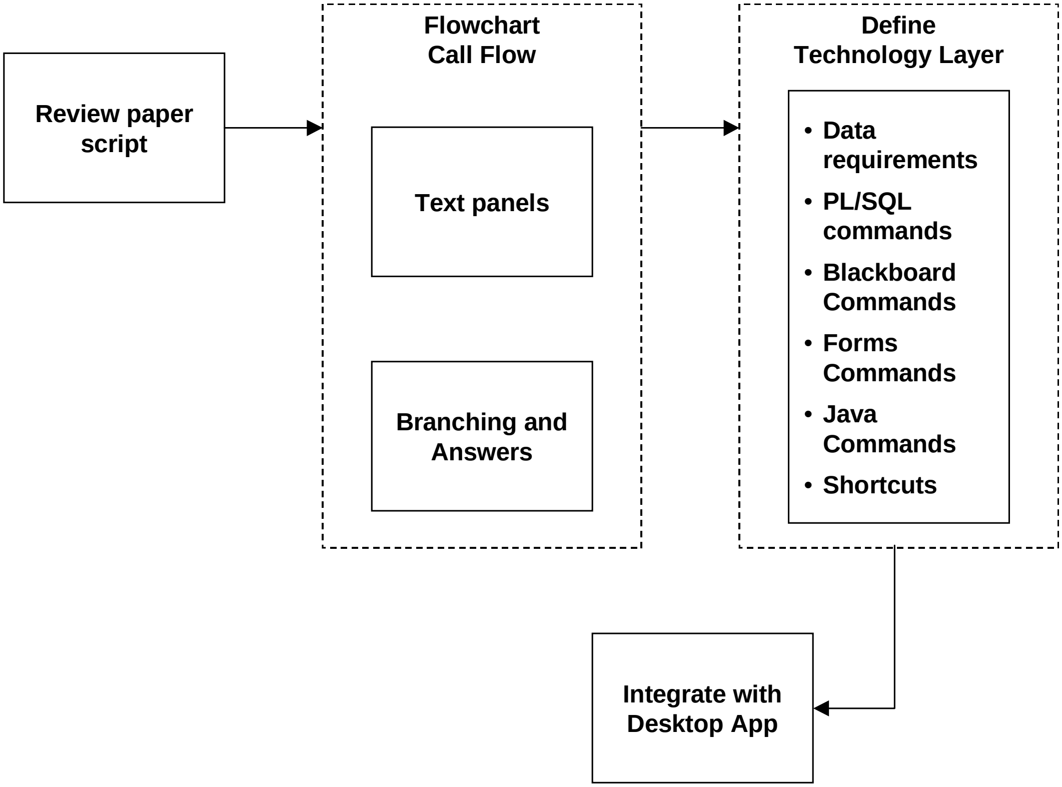


Figure 5. Processes involved in creating a script

Agent/Application Linkage

In order for an agent to use a script, the desktop application (Oracle TeleSales, Customer Support, etc.) must employ a method by which it determines which script to display to the agent. The script is most likely to be called based on one of the following:

- Agents are assigned to work on specific scripts and campaigns - more likely in an outbound telesales environment

- The application determines which script to run based on parameters in the customer database or on inbound DNIS or media type, or the script is hardcoded within the application.

Script Deployment Scripts are deployed from the Script Author by a process that compiles the script and loads it into the Oracle *8i* database, which also acts as the application server for the middle tier. This process is initiated from the Author tool by selecting **Deploy Script** from the **Tools** menu.

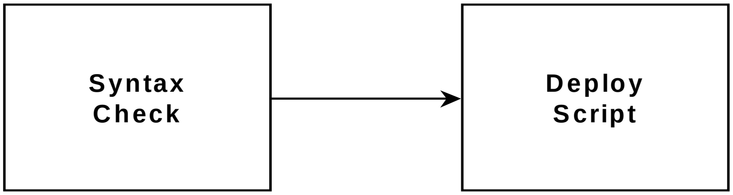


Figure 6. Selecting the* Deploy Script *command checks the syntax and, if successful, compiles the script and deploys it to the database

Typical Release Dependencies

Related Products and Components

The following is a discussion regarding related products and components that affect the implementation and ongoing operation of Oracle Scripting.

At the time of this writing, Oracle Scripting is supported (from a client perspective) on Windows NT, 95, 98, and Apple Macintosh systems running OS 9.0 or better. From a server perspective, Scripting relies on an Oracle *8i* database (version 8.1.6 or later) with the Scripting schema installed. The most common platforms for *8i* servers are Solaris and NT, although other platforms are also supported. For information on supported platforms, refer to the latest references. Information about supported Oracle 11i platforms can be found in the following two documents:

- Oracle 11i Concepts document
- Release 11.0.3 and Oracle *8i* Release 8.1.6 Interoperability Patch Notes

Note that data in Scripting applications can be accessed in tables stored in versions of Oracle database earlier than 8.1.6, although *8i* is required on the database server, and is the foundation of the CRM 11i suite of applications.

The Script Author does not require other Oracle Applications to be installed on the development machine, although it does need to be on a network and able to see the database server in order to deploy scripts. Scripting uses thin JDBC to deploy scripts to the database, bypassing the OCI layer and therefore obviating the requirement for an appropriately configured `TNSNAMES.ORA` file. Nonetheless, the development machine would benefit from such a configuration, in which Oracle Client is loaded on the machine and the developer can make SQL calls to the database, for example to query which scripts have been deployed to the database. In this way, SQL commands intended to be executed from Scripting can be verified in advance using SQL, or tested using the Script Engine, from the development machine.

As discussed above, the Script Engine is installed with the "Rapid Install" installation of CRM 11i. While the presence of the Engine is not required to develop scripts using Script Author, scripts will not be able to be deployed or executed without the appropriate CRM 11i installation, a functioning *8i* database, and network environment.

Following the installation steps mentioned in Setting Up Oracle Scripting > Prerequisites section above, no further processing should be required in order for scripts to execute, provided the user has an Oracle Applications login, is set up with database permissions to deploy a script, and the network and database are operational.

The Script Author is included with the Call Center Applications Installation CD. [Note: This CD has been replaced by an updated CRM Call Center Applications ARU. See Step 7 of *Installation* steps in this document.] Note, however, that no other call center applications are required for development of scripts using the Author.

Implementation starting point

At the start of your implementation, you will have an Oracle *8i* RDBMS with the Scripting schema installed.

Oracle CRM 11i "Rapid Install" is required, with CRM Family Pack 11.5.1A. For more information, refer to the detailed *Installation* steps in this document.

Considerations and constraints of interactions with other types of products

Since Oracle Scripting is a 100% Java application, it can be integrated with other Oracle CRM 11i products. Oracle Scripting provides an extensive set of APIs and events for external applications to monitor and control the flow of the script.

Scripting interacts with Forms 6.0 applications via the *Bean Area* (in Oracle Forms 4.5, this was called the *Java User Area).* The Forms-based application includes a Java wrapper that runs in the Bean Area and manages the communications between Scripting and the Forms-based application. This Java wrapper allows the application to call Scripting APIs and allows Scripting to pass data from the script to the application.

Scripting and Integration of Other CRM Applications

When we say that Scripting is "integrated with other CRM applications" we mean that any of the following activities can be performed:

- The application can call a script
- A Forms-based application can display one or more elements of the Scripting UI component beans (e.g., Panel Display bean and Progress Area bean; for others, see the discussion on *Script GUI components factory* in the Scripting API document referenced below) within a tabbed form within the CRM application (or a separate Scripting window). Thus, from a user point of view, it looks like the Script is actually another Form within the application
- The script can share data (via Data Sources) with the CRM 11i application
- The script can launch (or "pop") a specific form in the CRM 11i application based on the call context
- The application can pop a specific Group within a Script (e.g., to a certain predefined set of panels or other Scripting functions)
- The application can initiate events within the script (such as start and stop a script), using the published APIs

The degree of integration varies among the various CRM applications. Check the documentation for each appropriate application to determine to what degree they are integrated with Scripting.

Current Integrations between Scripting and Other CRM Applications

For Oracle CRM release 11.5.1 (also known as R1), Oracle Scripting is integrated with one application in the Oracle eBusiness Suite: the Oracle Customer Support module of Oracle TeleService. The degree of integration is limited at this time. There is a single Oracle Script (included with Oracle Customer Support out of the box) that, when invoked, performs a single function. The end result of the script is that a new service request is generated.

Note: It should be noted that this application is currently undergoing an enhancement so that any script can be invoked by the application; this enhancement (patch) is expected in the immediate future (prior to R2).

Oracle CRM release 11.5.2 (R2) will include additional Scripting integration with Forms-based applications. Oracle Banking Center will be integrated with ten banking function-related scripts out of the box. (Note that, if integration is required with legacy systems, this is a customization effort that would involve an OCS engagement.) Oracle TeleSales and Oracle Collections Center will also have the capability to call a script in R2, in a similar manner to the patched Customer Support application. No scripts are pre-integrated out-of-the-box for TeleSales or Collections Center.

Oracle iSupport is also integrated with Scripting to some degree. Note, however, that this integration is an entirely different matter from an architectural standpoint, since iSupport is not a Forms-based application. Essentially, iSupport provides a web GUI powered by the Scripting Engine, and to that degree, it is integrated.

On the horizon are additional integration efforts - more "canned" scripts for those products that provide them, and brand new integrations. Oracle Scripting has a published set of Scripting APIs that allow other applications to share information with Scripting. Some products in the eBusiness Suite must still be modified to function with Scripting.

Scripting and Customization Typically, a customer of scripting has included the customization of at least a single script to be developed by Oracle Consulting Services (OCS). Due to the nature of Scripting, these scripts are not considered to be supported by Oracle Support Services (OSS) unless (a) the issue is related to the actual product and not the customized script, or (b) such support has been arranged within the scope of a contract.

Modifying "Canned" Scripts Even if Scripting is integrated out-of-the-box with a specific application, customizations to those scripts included with the application (e.g., during an OCS engagement) are not considered to be supported unless such support is specifically contracted for. Modifying "canned" scripts for a given application is not without its own inherent risks. Each Oracle product development team understands the APIs, events, and data passed by its application and Scripting, and scripts to be provided with those products have been thoroughly tested. Consulting teams are not always privy to the same expertise, familiarity, and technical resources available to development teams of their respective products, and complications may effect scheduled delivery at great economic cost and with other contractual risks.

Related Documentation and Resources

The following related documentation may be of interest:

Installing Oracle Applications, Release 11i This guide provides instructions for managing the installation of Oracle Applications module.

Implementing CRM Applications This guide provides instructions for completing your installation of Oracle Customer Relationship Management (CRM) products. Refer to the section titled “Oracle Scripting.”

Oracle Call Center Applications Setup This guide covers the installation of the following Oracle Call Center and Telephony Applications components including the Oracle Scripting Author.

Oracle Scripting Concepts and Procedures This guide provides information and instructions to help you work effectively with Oracle Scripting.

Oracle8*i* Administrator’s Reference This reference includes information about creating and configuring databases, setting trace files, Configuring Java Option Connections, the IIOP listener for CORBA, etc.

Oracle8*i* Java Developer’s Guide This guide includes information about debugging server applications and schema object tools.

Oracle8*i* Enterprise JavaBeans and CORBA Developer’s Guide and Reference This guide includes information about how to set up the IIOP Listener and Dispatcher.

Setting Up Oracle Scripting

Overview

To test that Oracle Scripting has been successfully installed, we recommend that you first verify that you can create, compile, deploy and run a simple script (as described in step-by-step instructions below). Following that test we recommend deploying and testing a ready-made Regression Test script along with its associated Java files, as described in the *Launching a Script in the Test Form* section below. Following successful completion of those tasks, you can then copy the customized script (and associated files) developed specifically for this customer to the client machine hosting the Script Author and to the Scripting server, respectively. Instructions in the section titled *Writing and Deploying Custom Code to Enhance a Script* later in this document describe how you can deploy the custom code to the customer's database.

In order to test that Scripting has been installed successfully, you must follow these steps:

- Create a new script or open an existing script using the Script Author
- Compile the script
- Deploy the script to the 8.1.6 database
- Login as an agent or other script user and run the script
- Finally, follow procedures to copy the customized script into the Script Author, along with any associated files, and ensure that it can run successfully in the new environment

The next several sections will describe how to create a simple script, compile, and deploy it.

Prerequisites

The first part of any implementation is to verify that the product has been successfully installed. See the detailed steps and installation flowchart below.

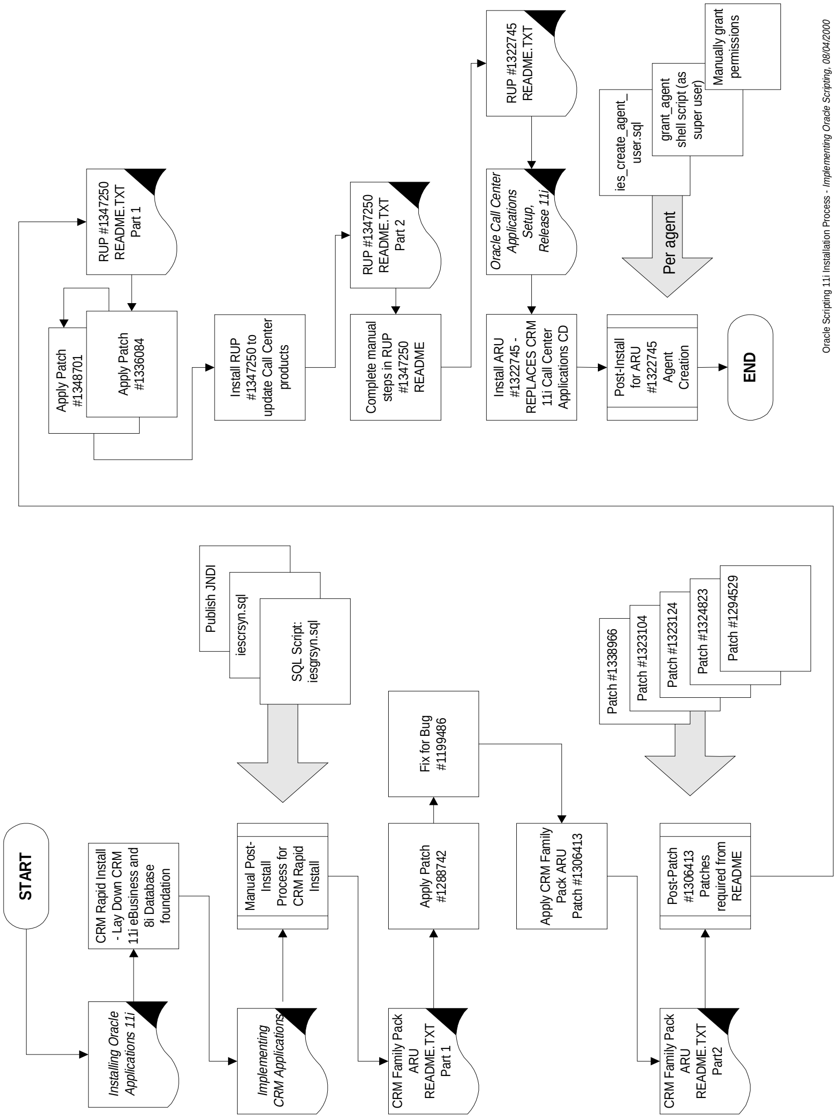

START
Installing Oracle Applications 11i
CRM Rapid Install - Lay Down CRM 11i eBusiness and 8i Database foundation
Implementing CRM Applications
Manual Post-Install Process for CRM Rapid Install
Publish JNDI
iescrsyn.sql
SQL Script: iesgrsyn.sql
CRM Family Pack ARU README.TXT Part 1
Apply Patch #1288742
Fix for Bug #1199486
Apply CRM Family Pack ARU Patch #1306413
CRM Family Pack ARU README.TXT Part2
Post-Patch #1306413 Patches required from README
Patch #1338966
Patch #1323104
Patch #1323124
Patch #1324823
Patch #1294529
RUP #1347250 README.TXT Part 1
Apply Patch #1336084
Apply Patch #1348701
Install RUP #1347250 to update Call Center products
RUP #1347250 README.TXT Part 2
Complete manual steps in RUP #1347250 README
RUP #1322745 README.TXT
Oracle Call Center Applications Setup, Release 11i
Install ARU #1322745 - REPLACES CRM 11i Call Center Applications CD
ies_create_agent_user.sql
grant_agent shell script (as super user)
Manually grant permissions
Per agent
Post-Install for ARU #1322745 Agent Creation
END
Oracle Scripting 11i Installation Process - Implementing Oracle Scripting, 08/04/2000

Figure 7. Flowchart of Oracle Scripting Installation

In the case of Oracle Scripting, there is a labyrinth of procedural steps, each with its own requirements, prerequisites, and dependencies. It is not the intention of the authors to duplicate any steps that have been carefully documented elsewhere; however, in light of the variety of patches that have been released, these detailed steps and flowchart are provided to ensure a successful installation and implementation.

Tools and Software Required

Before you start, the following are required:

- Physical access and associated privileges to the 11i/*8i* server
- Access to LoadJava utility
- Ability to run SQL scripts using tool such as SQL*Plus
- CRM Rapid install CD - Installs eBusiness Suite, 11i CRM Apps foundation and *8i* database

Patches Required

The following patches, rolled-up patches and automated release updates are required:

- Patch #1288742 - 11i Oracle Order Management Product Bundle
- Backported Patch for Server Technologies (a fix for bug #1199486)
- ARU #1306413 CRM Family Pack 1 ARU - patch to 11i apps family
- FND Patch #1338966
- Patch #1323104 - For Oracle Field Service/Laptop
- Patch # 1323124 - For Oracle Field Service/Palm Devices
- Patch # 1324823 - If Oracle Campaign Plus, Oracle IVR Integrator, Oracle Interaction Blending, Oracle Scripting, and/or Oracle eMail Center have been installed
- Patch #1294529 - iHelp patches for CRM Family Pack 1
- Patch #1336084 - For Oracle Telephony Manager upgrade
- Patch #1348701 - CRM Foundation Rollup Patch
  - RUP #1347250
  - ARU #1322745 - CRM Call Center Applications ARU

Patch sources:

Most or all of the patches listed below are available on the MetaLink web site, at http://metalink.oracle.com. If you do not have access to this web site, contact your Oracle Support representative, who will arrange to make patches available through FTP or another appropriate method.

Installation

Installing Oracle Scripting is an ten-part process:

1. **CRM "Rapid install" -** This install lays down the Oracle *8i* database, the CRM 11i schema, and the Scripting engine. Specific documentation exists for this installation; refer to the *Installing Oracle Applications 11i.*
2. **Manual Post Install process for CRM "Rapid install" -** These steps, described in the next section below, set up the Scripting Engine. This process is also documented in *Implementing CRM Applications*, under the section called *Oracle Scripting*.

> **Note:** This installation step is included below for the convenience of implementation; however, the process may change. Although it is the intention of the authors to update this *Implementation Guide*, it is your responsibility to refer to the *Implementing CRM Applications* document to verify which has a more recent publication date to ensure you are using the most updated information available.

3. **Family Pack 1 Automated Release Update (ARU) ReadMe (Part 1) -** Carefully review and begin to apply the steps in the `README.TXT` file that comes with the CRM Family Pack 1 ARU #1306413.

> **Note:** Note that after following the first two steps in this README.TXT file, the actual ARU #1306413 should then be installed.

Steps included in this ReadMe file to be applied *before ARU #1306413* are:

- Patch #1288742 - 11i Oracle Order Management Product Bundle
- Backported Patch for Server Technologies (a fix for bug #1199486)

4. **Apply CRM Family Pack 1 ARU -** After applying the appropriate prerequisite patches listed in the README.TXT file, apply the CRM Family Pack 1 ARU #1306413. This is a patch to the entire CRM 11i Applications family, and is available on MetaLink, which is accessed from http://metalink.oracle.com.

5. **Family Pack 1 Automated Release Update (ARU) ReadMe (Part 2) -**After applying ARU #1306413, *continue* applying the relevant patches listed in the `README.TXT` file accompanying CRM Family Pack 1 ARU. Steps included after applying ARU #1306413 include:

   - FND Patch #1338966
   - Patch #1323104 - For Oracle Field Service/Laptop
   - Patch # 1323124 - For Oracle Field Service/Palm Devices
   - Patch # 1324823 - If Oracle Campaign Plus, Oracle IVR Integrator, Oracle Interaction Blending, **Oracle Scripting**, and/or Oracle eMail Center have been installed
   - Patch #1294529 - iHelp patches for CRM Family Pack 1 (follow ReadMe instructions)

6. **RUP #1347250 ReadMe (Part 1) -** Follow the steps in the `README.TXT` file that comes with RUP #1347250. Steps included before applying RUP #1347250 include:

   - Patch #1336084 - For Oracle Telephony Manager upgrade
   - Patch #1348701 - CRM Foundation Rollup Patch

   Note: Note that this README.TXT file includes some post-installation steps that must be accomplished following the application of RUP #1347250. Please follow the instructions carefully. The post-installation step that requires Patch #1322745 may be disregarded as it is included in these instructions.

7. **Install RUP #1347250 -** Now that Oracle CRM 11i and the Oracle 8i database have been installed and updated, apply the Rolled-Up Patch (RUP) #1347250 to update Call Center products.

8. **RUP #1347250 ReadMe (Part 2) -** After applying RUP #1347250, continue following the steps in its accompanying README.TXT file.

9. **Apply CRM Call Center Applications ARU -** Install the CRM Call Center Applications Automated Release Update (ARU) #1322745. This patch is a ZIP

file that entirely replaces the Scripting Author installation from the CRM 11i Call Center Applications CD. The ARU installation process employs the Oracle Universal Installer to install the Oracle Scripting Author.

> **Note:** At the time of this writing, ARU #1322745 was not yet available on MetaLink and was only located on the tcpatch.us server. If you do not have access to this server, contact your Oracle Support representative, who will arrange to make the patch available through FTP or another appropriate method.

10. Perform the Post-Installation steps described below to set up Scripting agents.

Manual Post-Installation Processes and Troubleshooting

Oracle Scripting CRM Rapid Install Manual Post Install Process

The following steps need be applied only a single time on the CRM 11i server following the CRM Rapid Install. This process requires SQL*Plus (included in an Oracle Client installation) to execute the scripts against the server.

1. Run the `iesgrsyn.sql` script from SQL*Plus as SYSTEM. This grants applications users the permission to execute Java, which is required for a Scripting User:

   ```
   sqlplus SYSTEM/<SYSTEM password> @$IES_
   TOP/patch/115/sql/iesgrsyn.sql
   ```

2. Run the `iescrsyn.sql` script from SQL*Plus as APPS user. This provides synonyms for Scripting objects:

   ```
   sqlplus <APPS username>/<APPS password> @$IES_
   TOP/patch/115/sql/iescrsyn.sql
   ```

3. Publish the JNDI name for Oracle Applications server objects. JNDI is the protocol to call Java code in the database:

   ```
   publish -republish -user APPS -password APPS -service sess_
   iiop://localhost:<IIOP port>:<SID>
   /test/oracle/apps/ies/corba/common/Master
   oracle.apps.ies.corba.server.MasterImpl
   oracle.apps.ies.corba.common.MasterHelper
   ```

4. Note that a fourth step, included in *Implementing CRM Applications*, is not relevant in the context of this document, and refers to a CD which has been replaced by ARU #1322745.

Post-Installation Steps for Scripting Agent Creation

1. Following the installation of Scripting Author, copy the files in the server directory (including shellScripts and sqlScripts) to your 8.1.6 server machine.
2. Create one or more agent logins (one for each agent who will be using Scripting). To do this, run the script `ies_create_agent_user.sql` (found in the sqlScripts directory) in SQL*Plus for each agent login you wish to create.
3. Log into SQL*Plus with a super user account that has *create user* privileges.
4. Run the `grant_agent` shell script (found in the shellScripts directory) with each agent login that you created in step 2. Use the following syntax:

   ```
   grant_agent <agent login> <admin login> <admin password>
   <IIOP port> <SID>
   ```

   where

   - `<agent login>` is the login that was created in the previous steps
   - `<admin login>` and <admin password> should both be "APPS"
   - `<IIOP port>` is the port number that was used to configure the IIOP/CORBA listener
   - `<SID>` is the identifier for the database instance where Scripting is installed

5. Grant permissions to allow each agent login that you created in step 2 to execute Java.

   ```
   $ cd $JAVA_TOP/oracle/apps/ies/jar

   $ loadjava -grant <agent login> -user <APPS
   username>/<APPS password> -resolve -oracleresolver
   -synonym -definer -oci8 iescommn.jar

   $ loadjava -grant <agent login> -user <APPS
   username>/<APPS password> -resolve -oracleresolver
   -synonym -definer -oci8 iesservr.jar
   ```

Troubleshooting - Scripting Author 11.5.1 Installation

1. Scripting requires installation of the CRM Family Pack 11.5.1A. The Family Pack includes an extensive README.TXT file. Be sure to carefully review this document and do all of the steps that it specifies.
2. Scripting requires a patch to the 8.1.6 Applications Database. Contact Oracle Worldwide Support to receive the patch for bug #1199486. Note: this is documented in the Family Pack README.TXT file.
3. Verify that the Java option is turned on in the 8.1.6 Applications Database.
4. Verify that the IIOP listener for the 8.1.6 Applications Database is up and running and that the host name and port number for the listener is known.

Configure DB Settings for Expected Load

The following is the list of parameters in the 8.1.6 database's init.ora file that should be configured for scripting:

- `shared_pool_size` = 50 MB
- `java_pool_size` = (1 - 5) MB * number of clients [depending on size of script, 1 for small script, 5 for very large script]
- `sessions` = 50 + number of clients
- `processes` = 50 + number of clients
- `mts_servers` = number of CPU s
- `mts_max_servers` = <mts_servers> + 1
- remove `large_pool` if exists

Note: These are recommended settings, assuming only Scripting is being run. These numbers should be used as guidelines for extrapolating actual settings for a customer installation, with guidance from an experienced DBA who is familiar with the customer installation.

Troubleshooting - Solving Database Problems

Configuring IIOP Listener

- When configuring the IIOP Listener (performed by a DBA), ensure the host name of the machine is used, not just the IP address. The first time the listener is used, the IP address and hostname will be called, but subsequently *only the host* *name is used*. This is specified due to the method used in the setServerAPI listed in the API document.

Creating a Script

Overview

To confirm a successful installation, you must create a new script or copy the script already customized for the customer to a directory available to the Oracle Scripting Author.

Types of Files Used in Developing a Script

The typical script development process involves at least 4 different types of files:

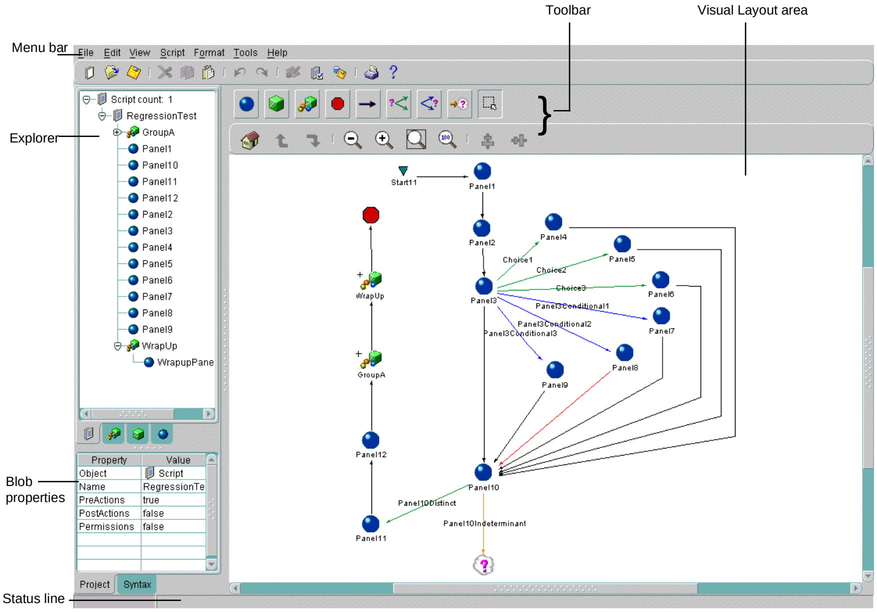
- **`.script` file -** Contains the metadata that has been created by the Oracle Scripting Author.
- **`.java` file -** Contains Java methods intended to add functionality to the desktop application. The Java methods contained in these Java files are needed for scripts that use shortcuts or which need complex logic. For example, any indeterminate branch in a script will have a Command that references a Java method. This is the actual file that is used to develop Java methods, from which a `.class` file is generated. The files can be created in the Java editor of choice.
- **`.class` file -** Contains compiled Java methods that are referenced by the `.script` file. The `.class` file(s) are associated from the Commands dialog within the Scripting Author. These files are required to be loaded into the JServer of the database (by themselves or, preferably, combined into JAR files) in order for the script to properly access and execute the methods. See the section *Deploying Code to the Database Using LoadJava*, below.
- **`.JAR` file -** Compressed Java Archive files. It is recommended to combine all relevant `.class` files referenced in the Scripting Author into one or more `.JAR` files prior to loading them into the JServer using LoadJava. This is done using the JAR command, part of the JDK.

If you are copying a script that has been customized for a customer by Oracle Consulting, you need to ensure that you have copied the `.script` file as well as any `.java` and `.class` files associated with it. Instructions in the section titled *Writing and Deploying Custom Code to Enhance a Script* later in this document describe how you can deploy the custom code to the customer's database.

The simplest script to create consists of a script with four objects placed on the canvas: two panels containing panel text (text that is displayed to a user), a group which is associated with the default Disconnect button in the Scripting Engine's GUI, and a terminal node. These objects would be connected with the appropriate branch type (in this case, default branching).

Overview of Scripting Author Environment

The Scripting Author is a development tool that allows script authors to use graphical tools to create and modify scripts. The entire script is represented as one or more graphs that show branching via visual symbols. Figure 8 shows some of the basic components of the development environment, although for an exhaustive list of all of the various icons and how to use them, refer to the *Oracle Scripting Concepts and Procedures* manual.



Property	Value
Object	Script
Name	RegressionTe
PreActions	true
PostActions	false
Permissions	false

Figure 8. The Script Author development environment

Steps to Creating a Script

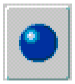
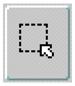
Creating a Simple Script

As noted earlier, the simplest way to determine that Oracle Scripting has been installed completely and successfully is to create, compile, and deploy a very simple script. The step-by-step procedures documented in the next several pages describe each step of the process for creating a two-panel script.

The script you will create by following the instructions in the next pages simply has an introductory panel which displays the text “Hello, World.” It then allows the user to press a button labelled Continue to access the second panel, which displays the text “Goodbye, World”. A Finish button at the end of the script terminates script processing.

In addition to the two panels (“Hello” and “Goodbye”), we also create a script group. A group is a sub-graph within a script which contains a set of logically related panels. One of the characteristics of a group is that it can act as the target for a jump to a specific location in a script. The group created in this exercise is a group that will always act as the target for the default Disconnect button (which is part of the standard Oracle Scripting default GUI). If you test this script as a stand-alone script, you can access the “Goodbye” panel by clicking on the Disconnect button upon entering the script.

Step Number	Required?	Script Creation Step-by-Step Description	Window Name(s)	AIW Reference
1.	Required	On the client PC which contains the Oracle Scripting Author, select the **Start** menu	Windows Desktop	
2.	Required	From the Programs listing, highlight **Oracle Scripting**, then select **Scripting Author**. This opens the Scripting Author development environment.	Windows Desktop	
3.	Required	From the Oracle Scripting **File** menu, select **New**. Oracle Scripting creates a new, empty script within the Script Author’s Visual Layout area. An icon called a Start node appears in the Visual Layout area, indicating that the new script has been started, and the Script Count in the Author’s Explorer area will be shown as 1.	Script Author	

Step Number	Required?	Script Creation Step-by-Step Description	Window Name(s)	AIW Reference
4.	Required	Click the **Panel Insertion** icon from the Toolbar to indicate you wish to insert a new panel.	 Script Author Visual layout	
5.	Required	Click ***once*** in the Visual Layout area somewhere below the Start node. This action drops a panel icon (shown as a blue ball) into the Visual Layout area.	Script Author Visual layout	
6.	Required	Click on the **Toggle Selection Mode** tool to turn off the Panel Insertion mode.		
7.	Required	**Right-click** on the panel that you just added to the script. The Script Author displays a drop down menu with the following options: Edit, Edit Blob Name, and Edit Blob properties. NOTE: If the dialog displayed shows only the options Edit and Edit Blob properties, then you have not correctly selected the Panel node. Left-click to close the menu and re-select the Panel node.	Script Author Visual Layout	
8.	Required	Select the **Edit Blob properties** option from the menu created by right-clicking on the new panel. The Script Author displays the Panel Properties dialog.	Panel Properties dialog	
9.	Required	In the **Panel Properties** dialog change the values of the following fields: Name - **Hello** Label - **Welcome**	Panel Properties dialog	
10.	Required	Click the **Apply** button in the Panel Properties dialog.	Panel Properties dialog	

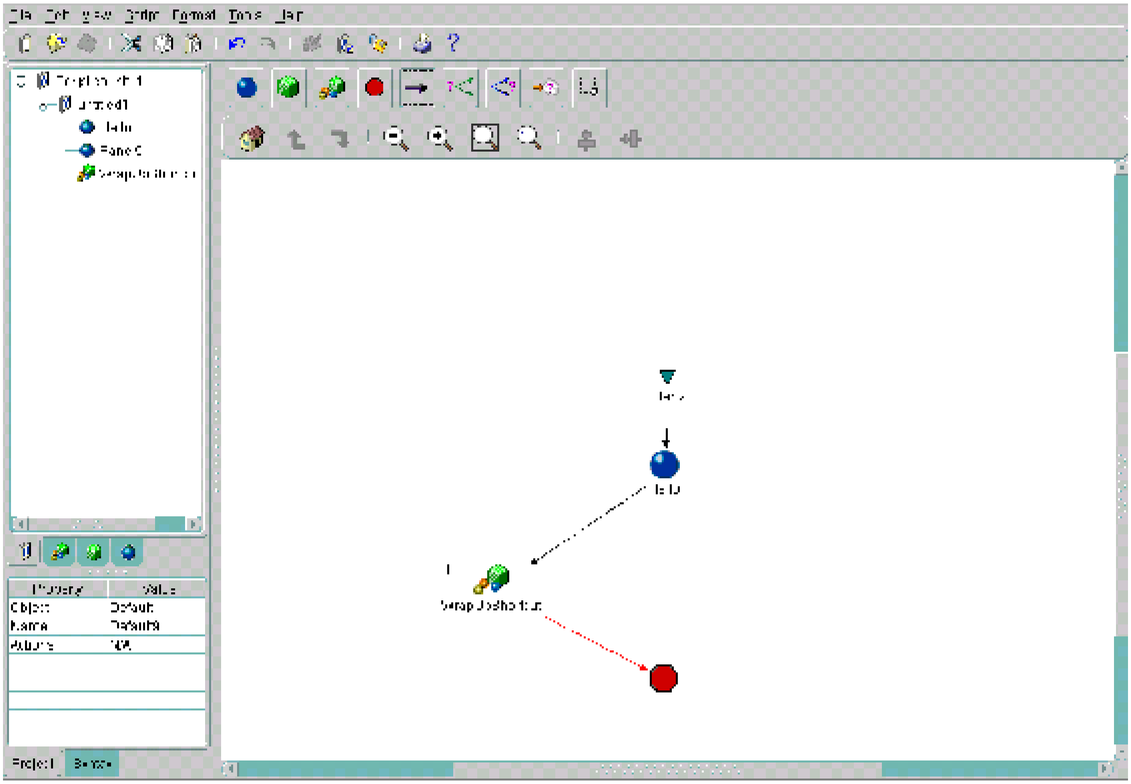
Step Number	Required?	Script Creation Step-by-Step Description	Window Name(s)	AIW Reference
11.	Required	Click on the **Answers** property displayed on the left pane of the Panel Properties dialog. The right pane of the dialog changes to display a text area which will show any answers you define for the script.	Panel Properties dialog	
12.	Required	Click the **Add** button which is immediately below the Answers text area. The Answer Entry dialog appears.	Panel Properties dialog	
13.	Required	Provide the following values for the fields in the Answer Entry dialog: Default for Distinct Branching - ***checkmark*** Name - **Continue_btn** UI Type - **Button** UI Label - **Continue** Click **OK** to close the dialog. The answer that you have defined now appears in the Answers text area in the Panel Properties dialog. It should appear as an entry with the name **Answer:Continue_btn (default)**. Note: If the word “default” does not appear in parenthesis, use the Edit button to re-enter the Answer Entry dialog and click on the Default for Distinct Branching checkbox. Apply, and then go to the next step.	Answer Entry dialog	
14.	Required	From the Panel Properties dialog, click on the **Edit Data Dictionary** button, which appears just below the Answers area. The application displays the Edit Data Dictionary dialog.	Panel Properties dialog	
15.	Required	Select the **Lookups** tab.	Edit Data Dictionary dialog	
16.	Required	On the Lookups tab, click on the **Specify Lookups** radio button.	Edit Data Dictionary dialog	
17.	Required	Click the **Add** button. The application displays the Lookup Entry dialog.	Lookup Entry dialog	

Step Number	Required?	Script Creation Step-by-Step Description	Window Name(s)	AIW Reference
18.	Required	In the Lookup Entry dialog, assign the following values to the fields: Display Value - **Continue** Value - **Continue** Click **OK** to close the Lookup Entry dialog and return to the Edit Data Dictionary dialog. Click **OK** to close the Edit Data Dictionary dialog and return to the Panel Properties dialog.	Lookup Entry dialog	
19.	Required	Click **OK** to close the Panel Properties dialog. The Visual Layout area should now show the newly-edit panel name.	Panel Properties dialog	
20.	Required	Click on the panel you have just created. When the markers appear around the panel to indicate that you have selected it, go to the **Tools** menu and select the **Panel Text Editor** option. This displays the Panel Text Editor.	Script Author, Tools menu	
21.	Required	In the Panel Text Editor dialog, click on the **Spoken Text Mode** icon. Then move your cursor to the Text entry area and type the following text: **Hello, World!**	t Panel Text Editor	
22.	Required	Go to the **File** menu in the Panel Text Editor dialog, and select **Exit**.	Panel Text Editor	
23.	Required	The Script Author will ask if you want to save the changes you have made to the Text for the panel. Click on the **Yes** button.	Panel Text Editor	
24.	Required	Now select the **Group Insertion** icon from the Toolbar.	Script Author Visual Layout	

Step Number	Required?	Script Creation Step-by-Step Description	Window Name(s)	AIW Reference
25.	Required	In the Visual Layout window, **click once somewhere below the panel** that you have already inserted. The Script Author will place a Group node into the script at the insertion point.	Script Author Visual Layout	
26.	Required	Click on the **Toggle Selection** button to turn off the Group insertion mode.	Script Author Visual Layout	
27.	Required	**Right-click** on the newly-added group. The Script Author will display a menu containing the following options: Edit, Edit Blob Name, Edit Blob Properties, and Show Subgraph. Select **Edit Blob Properties** to see the Group Properties dialog. NOTE: If the dialog displayed shows only the options Edit and Edit Blob properties, then you have not correctly selected the Group node. Left-click to close the menu and re-select the Group node.	Script Author Visual Layout	
28.	Required	Fill out the fields in the Group Properties dialog in the following manner: Name - **WrapUpShortcut** Comment - **Initiates an exit from script**	Group Properties dialog	
29.	Required	Click on the **Shortcut Action** property from the items on the left pane of the Group Properties dialog. The right pane is re-written and a new field appears. You must fill it out as follows: Shortcut Name - **WrapUpShortcut** Note that the Shortcut name must appear ***exactly as typed here, with upper and lowercase text and no spaces.*** During runtime, when the agent is using the compiled script, the Scripting Engine associates the Disconnect button with a group with the name WrapUpShortcut.	Group Properties dialog	
30.	Required	Click the **OK** button in the Group Properties dialog.	Group Properties dialog	

Step Number	Required?	Script Creation Step-by-Step Description	Window Name(s)	AIW Reference
31.	Required	From the Script Author Toolbar, select the **Default Branch Node** icon. The cursor turns into a cross-hairs symbol, indicating that you can begin drawing a line to define flow from one node to another within the Visual Layout area.	→ Script Author Visual Layout	
32.	Required	With the cursor displayed as a cross-hair, click and drag to **draw an arrow** between the Start icon and the panel called "Hello".	Script Author Visual Layout	
33.	Required	**Draw another default** arrow between the panel called "Hello" and the group called "WrapUpShortcut".	Script Author Visual Layout	
34.	Required	Select the **Termination Point Insertion Mode** icon from the Toolbar.	● Script Author Visual Layout	
35.	Required	**Click once** in the Visual Layout area somewhere below the WrapUpShortcut group icon. This will place a termination node into the script, which is required for each graph a script uses.	Script Author Visual Layout	
36.	Required	From the Script Author Toolbar, select the **Default Branch Node** icon. The cursor turns into a cross-hairs symbol, indicating that you can begin drawing a line to define flow from one node to another within the Visual Layout area.	→ Script Author Visual Layout	
37.	Required	With the cursor displayed as a cross-hairs, use click-and-drag techniques to **draw a line** between the WrapUpShortcut group icon to the Termination node.	Script Author Visual Layout	
38.	Required	At the File menu, select **Save** to save your work.	Script Author File Menu	

Step Number	Required?	Script Creation Step-by-Step Description	Window Name(s)	AIW Reference
39.	Required	In the Save menu, fill in the following: Look in - Select the directory where you wish to save the file. We suggest **a directory underneath the Script Author subdirectory** on the c: drive. Files of type - Select the default **Oracle CRM script file (.script)** File name: **implementation_script_1** Click the **Save** button to save the file in the selected directory. At this point, your script should look something like the shown in Figure 9.	Script Author Save dialog	



Property	Value
Object	Default
Name	Default9
Actions	N/A

Figure 9. The Visual Layout shows the main graph indicating default flow from the Start node to the Termination node

Creating the WrapUpShortcut Group

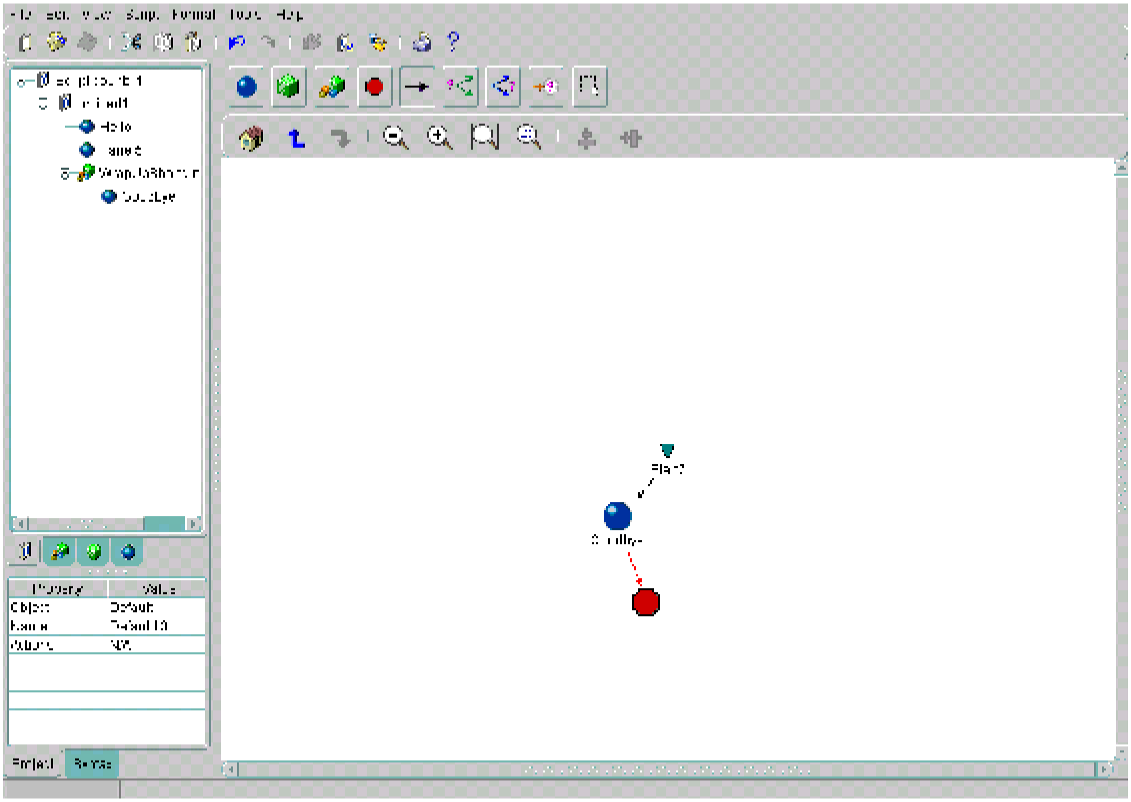
Step Number	Required?	Script Creation Step-by-Step Description	Window Name(s)	AIW Reference
1.	Required	Click on the WrapUpShortcut group on the canvas and select the **Go Down Into Child Graph** icon from the Toolbar. The Visual Layout area will immediately re-draw, shown a new graph (sub-graph) which is associated with the WrapUpShortcut group. As with the graph for the main portion of the script, a sub-group must have a Start node, at least 1 panel, and a Termination node. Branching must be used to “connect” these components with one another.	↴ Script Author Visual Layout	
2.	Required	Click the **Panel Insertion** icon from the Toolbar to indicate you wish to insert a new panel.	● Script Author Visual layout	
3.	Required	Click **once** in the Visual Layout area somewhere below the Start node. This will drop the panel icon (shown as a blue ball) into the Visual Layout area.	Script Author Visual layout	
4.	Required	Click on the **Toggle Selection Mode** tool to turn off the Panel Insertion mode.	⬚ Script Author Visual Layout	

Step Number	Required?	Script Creation Step-by-Step Description	Window Name(s)	AIW Reference
5.	Required	**Right-click** on the panel that you just added to the sub-graph. The Script Author displays a drop down menu with the following options: Edit, Edit Blob Name, and Edit Blob properties. NOTE: If the dialog displayed shows only the options Edit and Edit Blob properties, then you have not correctly selected the Panel node. Left-click to close the menu and re-select the Panel node.	Script Author Visual Layout	
6.	Required	Select the **Edit Blob properties** option from the menu created by right-clicking on the new panel. The Script Author displays the Panel Properties dialog.	Panel Properties dialog	
7.	Required	In the **Panel Properties** dialog change the values of the following fields: Name - **Goodbye** Label - **Goodbye**	Panel Properties dialog	
8.	Required	Click the **Apply** button in the Panel Properties dialog.	Panel Properties dialog	
9.	Required	Click on the **Answers** property displayed on the left pane of the Panel Properties dialog. The right pane of the dialog changes to display a text area which will show any answers you define for the script.	Panel Properties dialog	
10.	Required	Click the **Add** button which is immediately below the Answers text area. The Answer Entry dialog appears.	Panel Properties dialog	

Step Number	Required?	Script Creation Step-by-Step Description	Window Name(s)	AIW Reference
11.	Required	Provide the following values for the fields in the Answer Entry dialog: Default for Distinct Branching - ***checkmark*** Name - **Finish_btn** UI Type - **Button** UI Label - **Finish** Click **OK**. The answer that you have defined now appears in the Answers text area in the Panel Properties dialog. it should appear as an entry with the name Answer:Finish_btn (default). Note: If the word "default" does not appear in parenthesis, use the Edit button to re-enter the Answer Entry dialog and click on the **Default for Distinct Branching** checkbox. Apply and then continue to the next step.	Answer Entry dialog	
12.	Required	From the Panel Properties dialog, click on the **Edit Data Dictionary** button, which appears just below the Answers area. This displays the Edit Data Dictionary dialog.	Panel Properties dialog	
13.	Required	Select the **Lookups** tab.	Edit Data Dictionary dialog	
14.	Required	On the Lookups tab, click on the **Specify Lookups** radio button.	Edit Data Dictionary dialog	
15.	Required	Click the **Add** button. This displays the Lookup Entry dialog.	Lookup Entry dialog	
16.	Required	In the Lookup Entry dialog, assign the following values to the fields: Display Value - **Finish** Value - **Finish** Click **OK** to close the Lookup Entry dialog and return to the Edit Data Dictionary dialog. Click **OK** to close the Edit Data Dictionary dialog and return to the Panel Properties dialog.	Lookup Entry dialog	

Step Number	Required?	Script Creation Step-by-Step Description	Window Name(s)	AIW Reference
17.	Required	Click on the panel you have just created. When the markers appear around the panel to indicate that you have selected it, go to the **Tools** menu and select the **Panel Text Editor** option. This displays the Panel Text Editor.	Script Author, Tools menu	
18.	Required	In the Panel Text Editor dialog, click on the **Spoken Text Mode** icon. Then move your cursor to the Text entry area and type the following text: **Goodbye, World!**	t Panel Text Editor	
19.	Required	Go the **File** menu in the Panel Text Editor dialog, and select **Exit**.	Panel Text Editor	
20.	Required	The Script Author will ask if you want to save the changes you have made to the Text for the panel. Click on the **Yes** button.	Panel Text Editor	
21.	Required	Select the **Termination Point Insertion Mode** icon from the Toolbar.	● Script Author Visual Layout	
22.	Required	**Click once in the Visual Layout area** somewhere below the WrapUpShortcut group icon. This will place a termination node into the script, which is required for each graph a script uses.	Script Author Visual Layout	
23.	Required	From the Script Author Toolbar, select the **Default Branch Node** icon. The cursor turns into a cross-hairs symbol, indicating that you can begin drawing a line to define flow from one node to another within the Visual Layout area.	→ Script Author Visual Layout	
24.	Required	With the cursor displayed as a cross-hair, use click-and-drag techniques to **draw an arrow** from the Start icon to the panel called "Goodbye".	Script Author Visual layout	
25.	Required	**Draw another default branch** between the "Goodbye panel and the Termination node.		

Step Number	Required?	Script Creation Step-by-Step Description	Window Name(s)	AIW Reference
26.	Required	From the **File** Menu, select **Save** to save all changes to the script. Your WrapUpShortcut group should look something like the one shown in Figure 10.		



Property	Value
Object	Default
Name	Default ID
Action	N/A

Figure 10. The WrapUpShortcut sub-group also has Start and Termination nodes

Creating a Complex Script

Scripting is predominantly a single-developer tool. Early in development stages, multiple developers can theoretically work on different scripting portions of the script in the Author environment if the functionality of those portions is distinct. While Oracle Scripting has no team development tools in the R1 or R2 release, it does allow development teams to set up different "groups" (such as the WrapUpShortcut group in the example above) which contain discrete functions within the script as a whole. Each Script developer can work on a specific group, or set of groups, and then combine those groups to create an integrated script.

Challenges to Scripts Built in Parallel with Multiple Developers Note that, once certain groups or functions begin to call other groups or functions, the Scripting groups must be carefully integrated in order to syntax check the global script. This is also required in order to compile and/or deploy the script for testing purposes. This complicates development and increases the burden of communication on all team members. Parallel development is only recommended in cases where project management is experienced, and has the opportunity for frequent, detailed communication with team members on a daily (and sometimes hourly) basis. In such instances, careful, frequent backing up of individual groups (and tested, functioning integrated scripts) is an *absolute requirement.* Note that this may require substantial hard drive or network storage for the duration of the project.

Benefits and Challenges to Multiple Development Roles Multiple roles in development can further the early progress of a project. For example, one specialist can develop database hooks, another can create PL/SQL packages, a third can work primarily on custom Java methods, etc. However, the same caveats voiced above hold true in terms of integration or testing for each of these components.

Integrating Database Calls to a Script In Development In regard to integrating database components, additional challenges result. In isolation they can be developed and even tested, but when integrated into the main script, the development process is greatly complicated. From that point forth, all developers require a live database connection to continue development and testing, and any changes made to database hooks by that specialist must be carefully integrated and in fact thoroughly tested before development can be continued.

In such instances, it is recommended that development be allowed to proceed prior to final integration of database calls for as long as possible, to eliminate the additional time constraints on development personnel. Another approach is that a "dummy path" can be introduced into the script whereby most groups under development can be reached without accessing the panels or blocks that contain database procedures.

Unless such measures are taken, once integration of database calls has been completed, parallel development must cease, and a sole developer must work on the script at a time, increasing the amount of time required for script completion, increasing the burden on a single developer, and complicating further development and testing of the script.

Summary With careful management, individual developers' portions can be extricated from a tested script for continued development, but constant communication, testing, backing up, etc. are required if this approach is taken.

Compiling and Deploying a Script

Oracle Scripting Author has a built-in syntax-checking feature, available by selecting "Syntax Check" from the **Tools** menu. When you deploy a script (by selecting "Deploy Script" from the **Tools** menu), the syntax of the script is automatically checked.

If a script is deployed and passes the syntax check, the Script Author first compiles and then deploys the script to the database. Follow the instructions below to correctly define all of the parameters needed to compile and deploy the script you have just saved.

Step Number	Required?	Compiling and Deploying a Script	Window Name(s)	AIW Reference
1.	Required	From the Script Author's **File** menu, select **Script Properties**. The Script Properties dialog appears.	Script Author File menu	
2.	Required	Set the following properties: Name: **ImplementationTest** Language: **AMERICAN** Note that the value for Language must match an actual value from the NLS_LANGUAGE column of the FND_LANGUAGES table. For example, use "AMERICAN" for American English.	Script Properties dialog	
3.	Required	Click **OK** to close the Script Properties dialog.	Script Properties dialog	

Step Number	Required?	Compiling and Deploying a Script	Window Name(s)	AIW Reference
4.	Required	From the **Tools** menu, select **Deploy Script**. This displays the Deploy Script To dialog, which contains the following fields: Reuse an existing connection - **Do NOT click** Connection - **field is ignored** JDBC Driver- **oracle.jdbc.driver.OracleDriver** URL**- jdbc:oracle:thin:@<server machine name>:<TNS portnumber>:<SID>** User Name **- <Apps user DB login>** Password **- <Apps user DB password>** ***(Replace variables indicated by <x> with the appropriate value)***	Script Author Tools menu	

Troubleshooting - Compiling and Deploying a Script

1. The Oracle Script Author automatically performs a syntax check prior to compiling the script. If the script fails the syntax check, a status line at the bottom of the Scripting Author canvas indicates an error, and the contents of the Explorer navigation pane (to the left of the canvas) automatically populates with syntax-related information. This allows a script developer to debug the script.

Note: Clicking on the "Project" tab in the Explorer window will repopulate the Explorer window with the project view last displayed prior to failing the syntax check. The "Syntax" and "Project" tabs are both available whenever the Script Author is open. However, until the syntax is intentionally checked (or until a script fails the automatic syntax check associated with a "Deploy script" command), the only contents in the Syntax area of the Explorer will be a "Script unchecked" status message.

2. If a script passes the syntax check, but any problems are encountered loading and running scripts, review the *Installation* steps (*Prerequisites* under *Setting Up Oracle Scripting* section of this Implementation Guide).
3. If a JNDI error results, it is likely that the Master object has not been published. This step is detailed in *Oracle Scripting CRM Rapid Install Manual Post Install Process* (number 3).
4. If a script passes the syntax check, deploys to the database, and initially seems to launch but soon after encounters errors, ensure that any supporting Java code (in the form of `.class` files, which may be combined into one or more `.jar` files) have been loaded into the database. See the section *Deploying Code to the Database Using LoadJava,* below.
5. If a script passes the syntax check, deploys to the database, and launches before errors are encountered, and if the `.class` or `.jar` files are confirmed to be suitably loaded into the database, check in the Script Author that the `.class` file is correctly referenced in the specific Java command that calls the failing Java method.
    1. Check the Command type dialog where the Java method is referenced. The command type dropdown should be populated with the value "Java Command".
    2. The Command Info section of the Command type dialog should contain on the "Name" line the exact name (case and space sensitive) of the Java method in the `.class` file. The "Command" line under the "Name" line should contain the exact file name (case and space sensitive) of the Java `.class` file, followed immediately by two colons (::) and then the exact name of the method from the previous line.
    3. If the particular failing Java method (specified in the API document) is required to pass the Proxy bean (required in the interaction or connected state), ensure that "Proxy" has been added as a parameter in the Parameters section of the Command type dialog.
    4. If return values are specified in the Java method, ensure the associated information is entered in the "Return Value" portion of the Command type dialog.
6. If a script passes the syntax check, deploys to the database, and launches before errors are encountered, and if the `.class` or `.jar` files are confirmed to be suitably loaded into the database, and if the reference to the Java method is confirmed to be correct in the Script Author, the most likely source of errors is in the Java code.

1. If the Java code does not compile, readdress the issue in the Java development environment.
2. If the code does compile, more information will be required to debug a script and its Java code. In this event, refer to the trace files in the database.

7. Use Trace files to help debug code. Trace files are written to the database that log information regarding the status of Oracle Scripting, including its interaction with any associated Java methods. The DBA must activate tracing in the database. Refer to the Oracle *8i* database administration guide for details. In this way, script developers can debug Java and other errors reported by Scripting. This is the only reliable way to debug a script.

Writing and Deploying Custom Code to Enhance a Script

PL/SQL Code and Queries

Scripts can be customized in the Script Author with the use of SQL Commands, which call custom PL/SQL code during runtime. For more details on how to define and configure SQL Commands, see the *Oracle Scripting Concepts and Procedures* document.

The Script Author does not provide any facilities internally to write, compile, test, or deploy custom PL/SQL code. It is expected that this task is performed by an experienced PL/SQL developer, and that the custom PL/SQL code is developed and tested in the database before the script is configured to call it.

There is no special step needed to deploy PL/SQL code in order for Scripting to access it. As long as the PL/SQL is in the database, it can be accessed during script execution.

Java Code

Scripts can be customized in the Script Author with the use of Java Commands, which call custom Java code during runtime. For more details on how to define and configure Java Commands, see the *Oracle Scripting Concepts and Procedures* document.

The Script Author does not provide any facilities internally to write, compile, test, or deploy custom Java code. It is expected that this task is performed by an experience Java developer, with access to a full Java Integrated Development Environment (IDE) that is JDK 1.1.x or 1.2 compatible and has full editing, compiling, and debugging facilities.

Custom Java code is required to manipulate Scripting functionality via the published Java APIs. The Scripting API is encapsulated in the Proxy Bean object. The Engine includes the ability to pass a Proxy Bean object as a parameter to a Java Command object so that the custom code can call Scripting APIs.

Once the custom Java code is written and fully tested, it must be loaded into the JServer so that it can be accessed and run in the JServer during script execution, as described below.

Deploying Code to the Database Using LoadJava

Scripts developed in the Script Author are deployed from the authoring tool (using thin JDBC), which stores each script in the *8i* database schema at IES_DEPLOYED_SCRIPTS. Likewise, any custom Java classes referenced by Commands in the Script Author - and any third party libraries that the custom code calls - must also be loaded into the *8i* JServer. These additional files - typically `.zip` or `.jar` files containing custom code - are loaded into the *8i* JServer using LoadJava, as described below.

Limit what files are sent to the database. For example, note that the `.java` files are not required and should be excluded; otherwise, they will consume much-needed database storage space and may cause runtime errors. Also, it is recommended to combine all relevant `.class` files referenced in the Scripting Author into one or more `.JAR` files, which (among other reasons) will result in saving server space.

Step Number	Required?	Loading Java and Custom Code Using LoadJava	Window Name(s)	AIW Reference
1.	Required	Identify all files required to be loaded in the JServer. These include any custom Java classes (referenced by Commands in the Script Author) and any third- party libraries that the custom code calls.		
2.	Recommen ded	Use the jar command (part of JDK) to combine all relevant classes into one or more Jar files.		
3.	Required	Move the Jar (or other format) files to a directory on the database server machine using FTP or another appropriate method.		
4.	Required	Make sure `ORACLE_HOME` and `ORACLE_SID` environment variables are correctly set to the 8.1.6 database.		

Step Number	Required?	Loading Java and Custom Code Using LoadJava	Window Name(s)	AIW Reference
5.	Required	On the server at the command prompt, type the following: `loadjava -user <APPS username>/<APPS password> -resolve -oracleresolver -synonym -definer -oci8 <JAR_name>.jar`		
6.	Required	Repeat the above command for each type of file to be loaded, replacing the `<JAR_ name>.jar` variable with the appropriate format, e.g., `<zip_name>.zip`		

Troubleshooting - Deploying Custom Code to the Database

If a script loads deploys successfully to the database, but an associated Java file (or other file, as described above) fails to load with LoadJava, ensure the one-time process of executing the `iesgrsyn.sql` script (to grant permissions for applications users to execute Java) has been performed. This is referred to in general in Step 2 of the *Installation* steps, and addressed in detail as Step 1 of *Oracle Scripting CRM Rapid Install Manual Post Install Process*.

Proxy Server Settings

If Scripting is installed at a site that is protected by a firewall, the ICE Browser will not be able to access web pages outside the firewall unless its proxy server settings are configured properly.

The easiest way to do this is to make sure that the browser used to launch Forms has its proxy server settings (host name and port) configured properly (so it can access web pages outside the firewall), then select the “Use browser settings” checkbox in the JInitiator control panel.

Launching a Script

Overview

Launching a Script in an Oracle Form

A "Wrapper" - provided during the installation of any Forms-based product - allows Scripting to be embedded in a Form by wrapping the scripting GUI in Java code.

This serves to expose the script to the Oracle Form using the Bean area. This wrapper is housed on the web server (placed there by the installation of a Forms-based application), as are `iesclien.jar` , `iesservr.jar`, and `iescommn.jar`, three JAR files placed on the web server during the Scripting Engine installation and required for Oracle Scripting 11i to function. ***These JAR files and the wrapper should not be moved from this location, nor should they be modified.***

As part of the post-install processes for RUP # 1347250, the two JAR files required for the server (`iesservr.jar` and `iescommn.jar`) are *copied* via FTP to the database server, and then loaded in the JServer using LoadJava. The first time scripting code is executed, the wrapper and the two JAR files required for the client (`iesclien.jar` and `iescommn.jar`) are cached on the client machine (in a path such as `C:\Program Files\Oracle\JInitiator[`*`versionN`*`]`). Note that this may take several minutes, although subsequent launches should be nearly immediate.

Launching a Script in the Test Form

The "Rapid Install" includes a test Oracle form in which a deployed script can be launched for the purposes of testing that scripts are deployed appropriately, debugging scripts on a simple level, or to demonstrate Scripting's functionality.

Prerequisites

There are two prerequisites:

- Oracle Apps login with Scripting User responsibility
- Agent database login (created as described in the Post-Installation steps for Agent Creation)

Every application has one or more *responsibilities*; for example, for Call Centers using the Oracle Customer Support module of Oracle TeleService, each agent would have a Customer Support responsibility set up for him or herself. The systems administrator for Oracle CRM Applications 11i must associate the Scripting User responsibility (included with the "Rapid Install") with a particular applications login. That login should be used for the following test.

Step Number	Required?	Launching a Script in the Test Form	Window Name(s)	AIW Reference
1.	Recommended	Launch the JInitiator Control Panel from Start > Programs and check both checkboxes in the "Basic" tab. This will result in a Java Console which provides information regarding the status of Oracle Applications	Oracle JInitiator Properties > Basic Tab	
2.	Required	Log into Oracle Applications, using the login associated with the Scripting User responsibility		
3.	Required if multiple responsibilities	If multiple responsibilities are assigned to that login, select "Scripting User" from the resulting list.		
4.	Required	Required JAR files will cache on system, and an Oracle Form will open.	Oracle Applications - <SID Name>	
5.	Required	Fill out the information in the Script Information Panel.	Oracle Applications - <SID Name> Script Information	

Step Number	Required?	Launching a Script in the Test Form	Window Name(s)	AIW Reference
6.	Required	Input the following values to determine which script to execute: Script Name/Language: **ImplementationTest/AMERICAN** (or other value based on the script properties of the deployed script) Script Mode: **Three Tier** Display Mode: **<Your choice>** Embed Scripting Bean: **<Your choice>** For "Script Name/Language", type in the name of the deployed script. If a script was deployed using the above example, this value will be "ImplementationTest/AMERICAN" For "Display Mode", the value *true* will display a single panel of the script at a time; the value *false* will show several panels, with the active panel having focus and the inactive panels appearing dimmed out and using smaller display fonts. For "Embed Scripting Bean", selecting *true* will display the script on the right-hand side of that form; selecting *false* will display the script in a separate window.	Oracle Applications - <SID Name> Script Information	
7.	Required	Input the appropriate values to connect to the *8i* database into which the script was deployed: User Name: **<Agent Database Login>** Password: **<Agent Database Login Password>** JNDI Name: **/test/oracle/apps/ies/corba/common/Master** Host Name: **<Host Name of IIOP Listener>** Port Number: **<IIOP Port No.>** SID: **<SID>** Check with DBA if uncertain	Oracle Applications - <SID Name> Database Information	

Step Number	Required?	Launching a Script in the Test Form	Window Name(s)	AIW Reference
8.	Required	Click the "Start Scripting" button. The script will execute.		

Launching a Script in an Oracle CRM 11i Application

If using an Oracle CRM application integrated with Oracle Scripting, launch that application using the appropriate apps login, and then start scripting in the manner prescribed by that application. See the appropriate *Concepts and Procedures* manual.

Troubleshooting - Launching a Script

Commonly Encountered Difficulties with running Scripting 11.5.1

Problem: When running embedded within a Forms 6.0 form, the drop-down list of a drop-down control does not appear.

Solution: Contact Oracle Support for the patch to bug #1149726.

Problem: Deploying a script to the database with the Author fails.

Solution: The tablespace for the IES_DEPLOYED_SCRIPTS table may not be large enough. Particularly indicative is the scenario in which previous scripts were deployed to the database without difficulty. To resolve, increase the size of the tablespace (usually named USER_DATA) to an appropriate size (based on database usage and other installed applications). Consult your database documentation for details.

Problem: An error occurs while running the script: the Java console doesn’t have much error information to help debug the error condition.

Solution: Look at the trace files on the database server. They are located at:

```
$ORACLE_HOME/admin/$ORACLE_SID/bdump/*.trc
```

For information on setting trace files, refer to the Oracle *8i* database administration guide for details.

It may be useful to clear the contents of the directory, and reproduce the problem. This will leave only one trace file in the directory and no confusion as to which trace file is the one that documents the problem.

Performance Issues

If experiencing reduced performance subsequent to executing a script the first time, DBA should use the SQL command ANALYZE, which can optimize SQL queries using the Cost-Based Optimizer. Information on this utility is found in:

- the Oracle *8i* SQL reference
- the Oracle *8i* Designing and Tuning Performance reference
- Chapter 5 of the online Java Development Guide

References

For each of the documents referenced in *Performance Issues* above, search the Oracle Documentation Library.

Setting Profile Options

There are no system profile options in Oracle Scripting 11i.

System Profile Options

There are no system profile options in Oracle Scripting 11i.

Configuring and Testing Integration Points

Overview

E-Business products with which Oracle Scripting integrates

Oracle Scripting can be integrated with the Oracle Customer Support module of Oracle TeleService, as well as Oracle TeleSales, Oracle Banking Center, Oracle iSupport, and Oracle Collections Center. Please refer to the most recent documentation for each individual product listed above regarding integration methods and the degree of integration with Scripting.

Application Integration

Forms-based applications

Scripting is designed to work seamlessly with Forms-based applications. A few CRM 11i applications include integration with Scripting out of the box.

Custom Forms-based applications can also be integrated with Scripting. The Scripting UI includes several individual components that can be included in the application, via the Bean Area of Forms 6.0, as well as a Frame that includes all of the individual components in a default layout.

In addition, Scripting provides two sets of APIs for external applications to manipulate a running script. First, the Proxy Bean exposes APIs for manipulating the Scripting server object, and second, the Client Engine object allows an application to instantiate and manipulate the UI components of the Scripting product.

Note that the Applications login -- the login / password combination used for logging into Forms-based applications -- cannot be used to authenticate a Scripting session. The login used to authenticate a Scripting session must be an agent database login (created as described in the Post-Installation steps for Agent Creation).

CTI Integration

Scripting 11i will integrate with CTI. This takes three possible approaches:

1. Integrated with a CRM 11i business application (Customer Support, TeleSales, etc.) In this scenario, the business application handles the telephony integration. It communicates to Oracle Universal Work Queue (UWQ) and gets the script to do the appropriate thing at the appropriate time. Scripting doesn’t know or care about the telephony functions at all.
2. Scripting standalone integrated to CTI in CRM 11i. For this scenario, we have enabled Scripting to be CTI-enabled. That is not to say that it will be out of the box, because this integration is very implementation specific. However, we have ensured that Scripting 11i will integrate with UWQ and OTM 11i to take advantage of all the necessary telephony functionality. A consultant can come in and can follow our sample integration to build a specific integration for their client.
3. Scripting standalone integrated to other CTI suites. At the base level, we have attempted to make Scripting completely flexible so that it can be used in any implementation and doesn’t lose any useful functionality. To achieve this, we

have implemented two distinct APIs: one is non-GUI specific and allows any client to manipulate our server functionality (the Proxy Bean). The second is GUI specific (the Client Engine) and allows any custom application to manipulate our default GUI (and get access to the Proxy Bean for server manipulation). With these two sets of APIs, a consultant with Java programming experience can build a custom shell for his client that will integrate with the third party CTI vendor and will also embed and manipulate the Scripting functionality. For example, we have a simple API for starting a script: `startScript` takes two arguments, name of the script and language. The custom application needs to get the incoming call event from the third party CTI software and then call `startScript` to bring up the right script.

Testing integration points

There is no testing of the integration points due to the customization of the Scripting product to achieve integration.

Workflows

Where Oracle Scripting resides within the overall CRM 11i footprint

Oracle Scripting is part of the Call Center family of products. It can be considered a “tool” rather than an application, because it provides Authoring tools that allow trained users to customize their scripts for different agents, applications, and campaigns.

In fact, while there are some Best Practice scripts which represent generic credit card, collections, and telesales interactions and will be accessible “out of the box” with the TeleSales and TeleService (the Customer Support module), it is likely that most Oracle customers will customize the scripts they use for their specific interaction center requirements. This customization effort will always be associated with some Consulting engagement as part of the implementation of other CRM products. Note also that, as indicated in the *Current Integrations between Scripting and*

Other CRM Applications section above, such customized efforts are not supported by OSS unless contracted specifically.

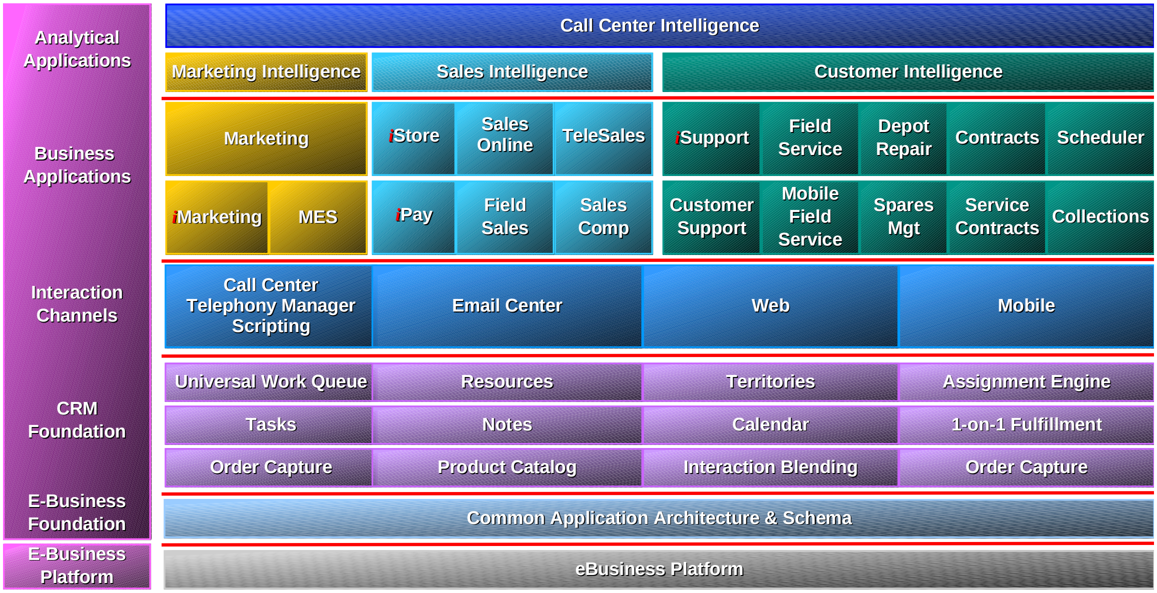


Figure 11. The overall CRM 11i footprint

There are no predefined workflows in Oracle Scripting. It is a standalone development tool that can be integrated with other eBusiness applications.

Best Practice Scripts

What Are Scripting Best Practices?

Best Practices Scripts provide off-the-shelf scripts and script templates that create successful call flows for agents on the telephone. Through the Oracle Best Practice Scripts, enterprises can immediately implement scripts into their production environments, effectively bringing the most successful approaches to customer interaction to the agent.

The Best Practice scripts exist as a library of re-usable modules focusing on specific subsets of the customer interaction. Best Practice Scripts also contain built-in data integration points to key database tables and views for querying and updating customer and account related information. This allows enterprises already using

Oracle ERP or CRM applications with immediate deployment using pre-built integration with their existing systems. For those enterprises who have their own external ERP applications, the Best Practice script templates contain pre-built modules which indicate where the "hooks" to the enterprises' applications can be put in place.

What is the Value to Oracle's Customers?

The Best Practice Scripts will be valuable for the following reasons:

- Provide out-of-the-box toolkit of best practice scripts and script templates, which can be easily moved into the call center production environment.
- Provide out-of-the-box integration with CRM applications.

For More Information on Best Practices Scripts

Refer to the *Oracle Scripting Concepts and Procedures Manual* for more information on Best Practice Scripts and their locations.

Converting and Inputting Existing Data

There is no converting or inputting of existing data into Scripting 11i.

Testing an Implementation Project

Overview

At minimum, regardless of whether a testing strategy and system test plans are developed, it is incumbent upon those implementing Oracle Scripting to perform two tests. The first is to deploy and then execute the script developed as part of the exercise listed in the *Creating a Script* section of this document (or another simple script). This is to ensure that the Script Author has been installed correctly and the server is configured correctly to allow a script developer to deploy a script to and execute it from the database.

Prerequisites

Prior to testing a scripting implementation, the environment must prove to be stable. The network and the database must be up and running, the "Rapid Install" must be accomplished, the "Family Pack" must be installed, and all PL/SQL packages tested prior to beginning testing. For more information on these steps, refer to the detailed *Installation* steps in the *Setting Up Oracle Scripting* section above.

Troubleshooting - Testing an Implementation Project

Scripting Variables When testing a scripting application, it is important to ensure each component is functioning on its own and in consonance with all others. From the standpoint of the script itself, one must test:

- The call guide or script (the document developed with Script Author)
- All custom Java methods developed to support the script
- All PL/SQL procedures attached as commands to panels, blocks, branches, etc.
- All blackboard commands
- All forms commands
- All constant commands
- All COM commands
- All delete actions

Environmental Variables One potential pitfall is the complexity of the scripting and Oracle applications environment. For example, even if all aspects of the script are functioning, the following are potential points of failure:

- Network
- Database server
- Web server
- PL/SQL package
- Granted user roles/privileges attached to logged in agent or scripting user (DBA level)

Considerations for Future Upgrade Paths

- Organizational considerations
- System considerations
- Application considerations

Acronym Glossary

API	Application Programming Interfaces
ARU	Automated Release Update (e.g., a patch)
CORBA	Common Object Request Broker Architecture
CRM	Customer Relationship Management
CTI	Computer-Telephony Integration
DBA	Database Administrator
GUI	Graphic User Interface
IG	Implementation Guide
IIOP	Internet Inter-ORB Protocol
JAR	Java Archive
JCK	Java Compatibility Kit
JDBC	Java Database Connectivity
JIT	Just-In-Time Compilation
JLS	Java Language Specification
JNDI	Java Naming and Directory Interface
JVM	Java Virtual Machine
OCI	Oracle Call Interface
OCS	Oracle Consulting Services
ORB	Object Request Broker, a feature of CORBA
OSS	Oracle Support Services
OTM	Oracle Telephony Manager
RUP	Rolled-Up Patch
SQLJ	Structured Query Language (SQL) embedded in Java
SysAdmin	Systems Administrator
UI	User Interface
UWQ	Oracle Universal Work Queue