INGRES

Ingres Best Practices: **Upgrading from OpenROAD 4.1 to OpenROAD 2006**

This document will guide you through the steps Ingres Corporation recommends to upgrade an OpenROAD 4.1 development environment to OpenROAD 2006.

Upgrading OpenROAD 4.1 to OpenROAD 2006 is straightforward and inexpensive, provided the upgrade is properly prepared for and carefully executed.

1 — Introduction

The recommendations in this document are based on extensive practical experience of such conversions, and have two objectives:

- To ensure that your upgrade is successful the first time
- To minimize time and effort required to upgrade

Upgrading OpenROAD 4.1 to OpenROAD 2006 is straightforward and inexpensive, provided the upgrade is properly prepared for and carefully executed. In most cases all that is required is the automated correction of known upgrade issues covered later in this document. In other cases, minimal code changes should suffice.

Note: If you are upgrading from OpenROAD 3.5 or 4.0, read this document in conjunction with *Ingres Best Practices: Upgrading from OpenROAD 3.5 to OpenROAD 2006* (currently being developed). That document addresses the additional conversion issues these prior versions entail.

2 — Approach

Most large organizations have many independent or semi-independent OpenROAD applications. Usually the same end users are running these applications. In other cases, independent groupings of end users are running applications. Some organizations also use multi-tier architectures using the OpenROAD Server. These situations raise the question of whether separate applications running different OpenROAD image versions can be upgraded independently.

Observe the following guidelines when you are considering selective upgrading of OpenROAD applications:

- Dual development environments are required for OpenROAD 4.1 and OpenROAD 2006.
- Running OpenROAD 4.1 images using OpenROAD 2006 runtime is not supported.
- Running OpenROAD 2006 images using OpenROAD 4.1 runtime is not supported.
- If you are using OpenROAD Server, you must upgrade it at the same time as the OpenROAD clients.

If your OpenROAD installation is complex or sizeable, these considerations emphasize the need to plan your upgrade project to OpenROAD 2006. The Appendix (page 16) details key considerations and steps in developing an upgrade plan.

3 — Upgrade Process

The upgrade process consists of the following basic phases:

Phase 1: Prepare for the upgrade.

Phase 2: Convert the applications, including resolving known issues.

Phase 3: Conduct acceptance testing.

Phase 4: Go live.

Each phase is detailed in the following sections.

Phase 1: Prepare for the Upgrade

We recommend the following steps to prepare for an upgrade:

1. Ensure a 4.1 test environment exists.
2. Ensure a 4.1 development environment exists.
3. Create a 2006 test environment.
4. Create a 2006 development environment.
5. Build and test your applications under 4.1.

The following sections explain each of these steps.

1. Ensure an existing 4.1 test environment exists.

Most live production systems already have a fully defined test area; you may clone this area for a 4.1 test environment. Or you could clone an active client environment.

Note: Do *not* clone a development environment. Experience has revealed that starting with a development environment wastes much time and effort during the upgrade. Reasons for this typically involve disparities in the environment settings and the application source code between the development and the live environments.

2. Make a 4.1 development environment available.

The upgrade process will highlight certain issues with your 4.1 applications. You need a working 4.1 development environment to investigate and document these issues.

The development environment must include access to the following:

- A database with a full copy of the source for all the existing OpenROAD applications
- Suitable data for the developer to exercise every significant business and maintenance path the application supports

3. Create a 2006 test environment.

The test area should correspond as closely as possible to the live deployment environment (and 4.1 test environment). The following checklist may be helpful in creating a proper test environment. These elements—or their absence—will impact the speed and success of your upgrade.

- OpenROAD installation (local or file-served). This includes:
  - Ingres and operating system environment variables
  - ingres\files area (especially config.dat and appedtt.ff, apped.ctb, and any variants)
  - ingres\w4glapps area (all referenced application images)
  - Any globals redefinition files
  - Any other files and file areas the applications reference (including 3GL library files, registered where necessary), with appropriate permissions
- The same test data that the 4.1 environment is working with, in a test database configured to match the live database (including any II_EMBED_SET or ING_SET settings). If you are copying the data using copydb, ensure that any contents of ii_stored_strings and ii_stored_bitmaps are copied as well.
- Ingres Net nodes (redirected to up-to-date test or training databases)
- Startup and other batch files (amended to reference the relevant databases, file systems, and so on, redirected as appropriate)
- Shortcuts (as these may contain explicit parameters or settings)
- User accounts, passwords, and permissions necessary to run the applications successfully

- Network connectivity (where it affects OpenROAD application use)
- Operating system, drivers, and settings (monitor, regional, and language settings in particular)
- Citrix and MainWin settings, where applicable
- Any referenced non-Ingres software, where its absence may prevent navigating to or displaying parts of the OpenROAD application
- A display monitor set to client specifications

Ensure that you can access this test environment through an account with sufficient privileges to issue commands such as ingsetenv.

4. Create a 2006 development environment.

You must specify the OpenROAD 2006 development environment fully and accurately—the previous checklist should prove helpful in this regard.

If possible, import a test or sample OpenROAD application, export it to your 2006 environment, and execute it from OpenROAD Workbench to confirm that your development environment provides the appropriate permissions, connectivity, settings, and so forth, listed previously. This may save considerable effort later.

Note: You will need OpenROAD 2006 plus patch 13047 or higher. For more information, see page 15.

5. Build and test the application under 4.1.

Building and testing the application under OpenROAD 4.1 is essential, but can be done quickly if you have prepared for them properly. This process flags discrepancies that may have crept in between the source application and the live application.

Test the application first in the development area.

Note: Any included images will need to be rebuilt. We recommend that you redefine all inclusions to be from the database during the conversion process; you can restore the image inclusions at the end.

Then build and deploy the imaged application in the test area, and repeat the testing.

The purpose of this testing is to establish whether the application code needs to be corrected, and whether the test environment is complete and valid. It does not need to do more at this stage.

Any problems encountered must be corrected or documented as being outside the scope of the upgrade before you continue.

Phase 2: Convert the Applications

Note: Do not proceed with the conversion until you have corrected or documented any issues with the existing applications. Ingres Corporation cannot support problems with your conversion unless you can demonstrate they are not pre-existent.

We recommend the following steps to convert your applications:

1. Copy the applications from the 4.1 environment to the 2006 environment.
2. Build and run the applications in OpenROAD 2006.
3. Identify and correct any remaining issues.
4. Update the "About" dialog.

The following sections detail each of these steps.

1. Copy the applications from the 4.1 environment to the 2006 environment.

We recommend the following procedure to upgrade each OpenROAD 4.1 application to 2006.

1. Using the 4.1 development environment, export the application using the exportapp utility from the 4.1 source code repository.
2. In your 2006 development environment, import the application into your 2006 source code repository using the importapp utility.
3. Export the application just imported from the 2006 source code repository, and re-import it into the same database using the -nreplace flag to overwrite what is already there.

   This re-export/re-import step is only a precaution. When a 4.1 application is imported into 2006, OpenROAD takes a number of extra steps to materialize the new attributes and other elements that have been added to the product. The second reimport will instead traverse the purely 2006 import path, ensuring that your application has been exposed to the full range of import processing before you start editing it.
4. Compile all the components in the application using the 2006 development environment.

2. Build and run the applications in OpenROAD 2006.

After you have created all the applications in your 2006 code repository, build and run them as follows:

1. Image any applications included as images in the correct build sequence (or better, change the image inclusions to be database inclusions: see the note about included images, page 4).
2. Run the application's principal threads to establish the validity of the test environment.

   The application should run correctly in all major respects: navigation, database interaction, functionality. If it does not run correctly, your 2006 development environment is probably incomplete; do not proceed until it is fully functional.
3. Run the application's principal threads to determine issues.

   These issues are usually visual ones, so it is important that this initial review of the application be reasonably comprehensive.
4. Identify and document issues, correcting immediately where appropriate.

   Whenever you encounter an issue, establish immediately in the 4.1 development environment whether it is a pre-existing issue and handle it accordingly.

   Where possible, correct issues that are likely to be frequent but can be corrected in an automated process before continuing. The section Known Issues (page 6) describes known conversion issues and how to correct them.
5. Correct any remaining issues that this initial review identified.

3. Identify and correct any remaining issues.

Run all the application's principal threads, this time touching all the application's displayed frames and functions. Identify and correct any outstanding issues.

4. Update the "About" dialog.

Remember to update your "About" dialog box if the upgrade affects its content.

Known Issues

There are a number of known issues when upgrading from OpenROAD 4.1 to 2006. Some of these issues relate to work-arounds or misuse of earlier versions of OpenROAD. These must be rectified when upgrading to OpenROAD 2006. For more information about when to correct these issues, see 2. Build and run the applications in OpenROAD 2006 (page 6).

Incorrect FP_CLEAR field setting

This issue is widespread, and will need to be corrected in most applications. It typically manifests as frames having areas with the wrong color.

ISSUE: In versions prior to OpenROAD 2006 the BgPattern attribute could be set to FP_CLEAR to make the field background clear or translucent. The use of FP_CLEAR was supported only for ShapeFields, although it also worked correctly for FreeTrim fields. For all other field types its behavior was identical to BgPattern=FP_SOLID. In OpenROAD 2006 all FormFields are documented as honoring the FP_CLEAR setting.

As a result of experimentation by developers, some field types have been left set incorrectly with BgPattern=FP_CLEAR.

RESOLUTION: Update all occurrences of fields (other than ShapeFields and FreeTrim) from BgPattern= FP_CLEAR to BgPattern=FP_SOLID.

To achieve this, Ingres Corporation provides the PropertyChanger tool. To request the PropertyChanger tool, contact Ingres Support at http://www.ingres.com/services/enterprise-support.php.

To use the PropertyChanger tool on your applications

1. Download the PropertyChanger tool and unzip it.
2. Move the SourceElements.img file into your II_SYSTEM\ingres\w4glapps directory.
3. Import the PropertyChanger.exp file as a new application into your OpenROAD 2006 code repository database.

Note: Before updating any of your applications, we recommend that you back them up by either exporting the affected applications or taking a checkpoint of the OpenROAD 2006 code repository database.

To change BgPattern settings using the PropertyChanger

1. In the PropertyChanger application, examine the PropertyChanger component behavior defined for FP_CLEAR, and confirm that it takes the required action. The following is the default code provided to make this change:

```
/*
** Change all fields that have FP_CLEAR
** (except shapes or freetrim)
*/
IF fieldToChange.IsA(FreeTrim) = TRUE THEN
ELSEIF fieldToChange.IsA(ShapeField) = TRUE THEN
ELSEIF fieldToChange.IsA(MenuField) = TRUE THEN
ELSEIF FormField(fieldToChange).BgPattern = FP_CLEAR THEN

 FormField(fieldToChange).BgPattern = FP_SOLID;

ENDIF;
```

   Modify this code as necessary to meet your requirements.

2. From a command window run the following command:

   `w4gldev rundbapp` *`yourdatabasename`* `propertychanger`

   An Upgrade Assistant will be displayed.

3. Using the assistant, do the following:

   a. Select which applications you are correcting.

   b. Specify that you want to change fields only.

   c. Specify that you want to correct FP_CLEAR.

   d. Specify the suffix to use if you want the assistant to make backup copies of your existing applications. Leave this blank if you do not want backup copies to be made (for example, if you have backed up your applications elsewhere).

   e. Click Upgrade to make the changes.

   Note: You may want to save the PropertyChanger output to help with later acceptance testing (see Phase 3: Conduct Acceptance Testing, page 13).

4. Compile and run the applications to confirm that the FP_CLEAR problems are corrected and these changes have caused no new issues.

   If you created backup copies using the Upgrade Assistant, delete them after you are satisfied that the changes are correct.

Note: Review the next issue, Incorrect field colors (page 9) to determine whether you need to use PropertyChanger again. If not, remove the SourceElements.img imagefile and the imported PropertyChanger application.

Incorrect field colors

If you are using a site-specific color definition file for OpenROAD, this issue may not affect your applications. This issue typically manifests itself as display areas showing the "wrong gray" when the developer starts adjusting field positions, margins, and transparency.

ISSUE: OpenROAD changed the color definitions it used between OpenROAD 3.5 and OpenROAD 4.0. These definitions have not changed since. For systems already on OpenROAD 4.x, visible color issues will have been resolved already. However, a potential problem remains with the varieties of gray.

In applications developed initially in OpenROAD 3.x, CC_PALE_GRAY and CC_LIGHT_GRAY were identical, and many fields were set to one or the other indiscriminately. When those applications were upgraded to OpenROAD 4.x, fields not visible to the developer—generally composite fields—were not always corrected.

The definition of CC_SYS_BTNFACE also changed over time: on Windows NT 4 the buttonface color was defined as rgb(192,192,192). It is now rgb(212,208,200).

In the upgrade to OpenROAD 2006, fields that have these color problems may not be visible initially. They may become evident only when changing to a background of FP_CLEAR or when changes are made to composite field margins.

Note: Any references to CC_LIGHT_GRAY (or CC_SYS_BTNFACE) in the original 3.5 code logic that might be broken by making the following change should have been found and resolved during the previous upgrade to 4.x. If you suspect that it has not been fixed, do not make the following change.

RESOLUTION: You can standardize color attributes using the PropertyChanger tool. The following procedure assumes that you have already loaded the PropertyChanger tool into your OpenROAD project (see To use the PropertyChanger tool on your applications, page 7).

Note: Before updating any of your applications, we recommend that you back them up by either exporting the affected applications or taking a checkpoint of the OpenROAD 2006 code repository database.

To change field colors using the PropertyChanger

1. In the application, examine the PropertyChanger component behavior defined for CC_LIGHT_GRAY, CC_PALE_GRAY, and CC_SYS_BTNFACE, and confirm that it takes the action you require. The following is the default code provided to make this change:

```
/*
** Change all fields that have CC_LIGHT_GRAY to
** CC_PALE_GRAY.
** (but ensure tablefield reset does not override its
** columns, tablebody, tableheader)
**
** Change all fields that have
** CC_SYS_BTNFACE to CC_PALE_GRAY.
** (disabled - only enable if the platform you are currently
** deployed on uses rgb(192,192,192) and you want to preserve
** that shade of gray)
*/
IF fieldToChange.BgColor != CC_LIGHT_GRAY
/* AND fieldToChange.BgColor != CC_SYS_BTNFACE */
 THEN
 /* do nothing */;

ELSEIF fieldToChange.IsA(TableField) != TRUE THEN
 fieldToChange.BgColor = CC_PALE_GRAY;

ELSE
 tbl = fieldToChange;
 copiedTbl = tbl.Duplicate();

 tbl.BgColor = CC_PALE_GRAY;

 tbl.TableBody.BgColor = copiedTbl.TableBody.BgColor;
 IF tbl.TableHeader IS NOT NULL THEN
 tbl.TableHeader.BgColor = copiedTbl.TableHeader.BgColor;
 ENDIF;

 FOR i = 1 TO tbl.TableBody.ChildFields.LastRow DO
 col = tbl.TableBody.ChildFields[i];
 copiedCol = copiedTbl.TableBody.ChildFields[i];
 col.BgColor = copiedCol.BgColor;
 IF col.TitleTrim IS NOT NULL THEN
  col.TitleTrim.BgColor = copiedCol.TitleTrim.BgColor;
 ENDIF;
 ENDFOR;

ENDIF;
```

   Modify this code as necessary to meet your requirements.

2. From a command window run the following command:

   ```
   w4gldev rundbapp yourdatabasename propertychanger
   ```

   An Upgrade Assistant will display.

3. Using the assistant, do the following:

   a. Select which applications you are correcting.

b. Specify that you want to change fields only.

c. Specify that you want to correct Gray Color Settings.

d. Specify the suffix to use if you want the assistant to make backup copies of your existing applications. Leave this blank if you do not want backup copies to be made (for example, if you have backed up your applications elsewhere).

e. Click Upgrade to make the changes.

Note: You may want to save the PropertyChanger output to help with later acceptance testing (see Phase 3: Conduct Acceptance Testing, page 13).

4. Compile and run the applications to confirm that the color problems have been corrected and these changes have caused no new issues.

   If you created backup copies using the Upgrade Assistant, delete them after you are satisfied that the changes are correct.

Note: When you are done using them, remove the SourceElements.img imagefile and the imported PropertyChanger application from your OpenROAD project.

Incorrect truncation of date-times

This issue manifests as incorrect truncation of date-times. It is therefore difficult to detect, but can be anticipated and avoided as detailed below.

This issue does not affect your OpenROAD applications or require changes to them. However, it may affect the rollout of the OpenROAD 2006 upgrade to client machines in OpenROAD client/OpenROAD Server architectures, if you are upgrading from versions prior to OpenROAD 4.1 Service Pack 3.

ISSUE: OpenROAD 4.1 SP3 and OpenROAD 2006 use different versions of the Ingres shared libraries and Ingres Net than OpenROAD 4.1 SP2 or previous versions. The newer libraries and Ingres Net contain changes to how they handle certain data types and functions to conform to ANSI database standards. In particular, the handling of "date without time" has changed.

Because OpenROAD is built and installed with a particular version of Ingres Net, OpenROAD clients and application servers can only communicate dates without time to each other if they are both pre-OpenROAD 4.1 SP3, or both post-OpenROAD 4.1 SP2.

RESOLUTION: All OpenROAD clients and servers that inter-communicate must be upgraded together, and rolled back together.

Remember to preserve an application maintenance capability in both current and target OpenROAD versions, so that both rollout and rollback can be handled successfully.

Change in handling of synchronous ActiveX events

This issue affects only applications using ActiveX components.

It does not affect those upgrading from OpenROAD 4.1 SP3.

It also does not affect applications that have no ON EXTCLASSEVENT code blocks.

This issue manifests in a variety of ways, and unpredictably. It is therefore difficult to detect, but can be anticipated and avoided as detailed below.

ISSUE: A change introduced in OpenROAD 4.1 SP3 resulted in OpenROAD 2006 handling ActiveX events differently from OpenROAD 4.1 SP2 or previous releases.

Prior to OpenROAD 4.1 SP3, when a frame invoked an ActiveX component that generated events (Internet Explorer generates many progress-related events), those events went to the end of the OpenROAD queue. Although this was simple to handle, in many situations the information arrived too late to be useful.

From OpenROAD 4.1 SP3 and forward, these events were handled immediately when they were received, similar to .NET and Visual Basic. When such an event is received the next statement executed is the first code statement associated with that event. This means that where ActiveX events are involved, developers can no longer rely on the "single thread principle" that no event will start until the current event has completed.

The following scenarios can result in a change of behavior when upgrading to OpenROAD 2006. Changes will be necessary if:

- The code in an ActiveX event block depends on OpenROAD code having been executed since the ActiveX component was invoked
- The code in an ActiveX event block itself invokes, directly or indirectly, an ActiveX component whose events trigger some other OpenROAD code

RESOLUTION: Identify all such cases in your applications. A simple code change corrects most cases by effectively restoring the behavior prior to the upgrade.

Split the ON EXTCLASSEVENT code block into two code blocks by inserting extra code into the code block, immediately after the BEGIN (or "{") statement, as illustrated in the following example.

- Code BEFORE upgrade:

```
ON EXTCLASSEVENT StatusTextChange
(
 p_v_text = VARCHAR(256) NOT NULL;
)=
BEGIN
 … do stuff …
END;
```

- Code AFTER upgrade:

```
ON EXTCLASSEVENT StatusTextChange
(
 p_v_text = VARCHAR(256) NOT NULL;
)=
BEGIN
 CurFrame.SendUserEvent('XEvent StatusTextChange',p_v_text=p_v_text);
END;

ON USEREVENT 'XEvent StatusTextChange'
(
 p_v_text = VARCHAR(256) NOT NULL;
)=
BEGIN
 … do stuff …
END;
```

UNIX installation error

This issue affects only applications running on UNIX operating systems.

This issue manifests as an OpenROAD installation error.

ISSUE: OpenROAD 2006 is built on MainWin 5.1.1. This version of MainWin has different core services and system requirements than OpenROAD 4.1, which is built on MainWin 5.0.2. These differences will affect installation both on rollout and rollback.

RESOLUTION: Ensure that you have established and built the uninstallation and installation requirements into your rollout processing.

Phase 3: Conduct Acceptance Testing

Individual organizations will have their own acceptance testing mechanisms for their OpenROAD applications. Apply these to the upgraded environment.

Important! It is critical that you perform the pre-testing described in 5. Build and test the application under 4.1, page 4 ("Any problems encountered must be corrected or documented as being outside the scope of the upgrade before you continue"), *before* doing the upgrade, so that you can isolate genuine upgrade issues and investigate them.

You may need to further specify testing for transparency and color issues. If, during the issue identification process, you recorded a variety of instances where FP_CLEAR or color settings were causing visual discrepancies in OpenROAD 2006, you can use these for the acceptance testing. PropertyChanger tool output includes a list of all named fields in all frames corrected, and you can use this for such a record. For more information, see Incorrect FP_CLEAR field setting (page 7).

If the ActiveX event issue applies (see Change in Handling of Synchronous ActiveX events, page 12), test the updated code carefully, especially if you used solutions other than those recommended.

Phase 4: Go Live

Ingres does not support using OpenROAD 2006 to run OpenROAD 4.1 application images. You must recompile and reimage your applications using OpenROAD 2006, then redeploy them to the client installations.

The resolution of certain issues may involve changes to files in II_SYSTEM\ingres\files, for example color or font changes, or changes to Ingres environment variables, in upgraded installations.

Individual organizations will have their own rollout mechanisms for OpenROAD applications. These must take into account the following points:

- Ingres does not currently support the installation of multiple OpenROAD instances on a single computer (unless individualized within virtual machines). All applications that need to coexist on a client computer must be upgraded together.
- If you run an OpenROAD client/OpenROAD Server architecture and are currently using OpenROAD 4.1 SP2 or a previous version, the "date without time" handling introduced in OpenROAD 4.1 SP3 will affect your applications. (See Incorrect truncation of date-times, page 11.) In this situation, all clients and all servers using this architecture must be upgraded or rolled back together.
- OpenROAD applications on UNIX require a different version of MainWin from those on 4.1, with different system requirements. (See UNIX installation error, page 13.)

In every other respect, a live environment appropriate for OpenROAD 4.1 deployment is appropriate for OpenROAD 2006 deployment.

Installing OpenROAD 2006 does not require uninstalling OpenROAD 4.1. However, for rollback you will need to uninstall OpenROAD 2006 (using InstallShield) before reinstalling OpenROAD 4.1 on the client.

To apply the recommended patch (13047 or higher) after OpenROAD 2006 is installed, you must stop the Ingres installation, carry out the patch instructions, and restart Ingres afterward. To receive the patch, contact Ingres Support at http://www.ingres.com/services/enterprise-support.php.

Other considerations, such as archiving the means to recreate a fully functioning 4.1 environment so that you can recover after an unsuccessful rollout, have already been highlighted.

Appendix — Planning Your Upgrade

The process of upgrading OpenROAD version 4.1 to 2006 may simply involve upgrading the software and performing some basic tests. However, for a more complex or sizeable OpenROAD installation we recommend that you take a more considered approach and create a Project Plan with which to control the upgrade.

This section suggests areas to consider when planning an upgrade. It is not intended as a definitive upgrade plan, but as a guide to creating one. Your actual plan will depend on the scale, complexity, and methodologies of the installation and your business.

The process of developing an upgrade plan consists of the following basic stages:

1. Initiation
2. Testing
3. Backout strategy
4. Scheduling the upgrade
5. Live rollout
6. Creating the plan

1. Initiation

Initiation is the data- or fact-gathering stage and is concerned with investigating and documenting the current installation, which includes identifying all existing instances of OpenROAD, and gathering and documenting the requirements for the new environments.

Record any settings relevant to the existing OpenROAD installation. You can find the basis for a worksheet in 3. Create a 2006 test environment (page 3).

You should determine and establish the roles and necessary skills required for the upgrade and ascertain the availability of the necessary people on the upgrade team.

Team members should establish and agree upon any expectations or mandatory requirements for the upgrade. For example, your business may have set certain performance targets or expectations as to when the upgrade should be in place. Incorporate these into your plan.

After this initiation process, the scope of the upgrade should be clear. Add all these elements to the plan as outlined in 6. Plan Creation (page 18).

2. Testing

Upgrading from one version of software to another introduces change. To ensure a trouble-free transition and to minimize risk, testing is essential. The amount of testing necessary depends on the scale of your installation and the number of applications.

When upgrading, the software is updated and any existing OpenROAD application images re-imaged under the new version. Your test plan should include testing the following:

- The software upgrade process
- The applications that will use the upgraded software
- The rollout process

Conduct testing until all aspects of the upgrade work from start to finish without error.

Most applications will have an associated system test plan. It is a good idea to run this plan for all applications to ensure that the new version of the software and application has introduced no new issues. If no system test plan is available, then you should create one that will at least verify that the basic functionality of an application has not been affected adversely.

Consider the amount of user acceptance testing required, if any. In principle this should not be necessary; however, it depends on how much business knowledge is held by the team doing the system test. If the test team's knowledge is considered inadequate to verify the basic application functionality, a stage of user acceptance testing will be required.

3. Backout Strategy

Even the "best laid plans" can go awry. Your upgrade plan should consider what to do if problems are encountered.

The plan should consider the severity of any potential problems and recommend appropriate action. For example, in a worst case scenario it might be necessary to back out the upgrade entirely. The strategy should also consider how long the upgrade is in place before it can be considered irreversible.

To back out an upgrade it may be necessary to have secured the original version of the system. The plan should include all data to be backed up, including transactional, static, and configuration data, and the methods employed to do so.

4. Upgrade Scheduling

An OpenROAD installation might include development, test, system test, user acceptance testing, training, and production environments. Consider how these are to be upgraded and in what sequence to ensure that business-driven changes can still be supported during the upgrade process, without requiring re-work or re-application of bug fixes, changes, and enhancements. It may be necessary to maintain development and testing environments at different versions of OpenROAD to allow for this.

Plan and schedule the timing of each upgrade to minimize risk and disruption.

5. Live Rollout

Estimate the time required and availability of necessary team members during rollout so that you can communicate to your business any necessary downtime for the upgrade in the production environment. You can then agree upon a "go live" date.

You should agree upon the level of testing following rollout that is necessary to prove the live upgrade was successful. Then incorporate this information into the final rollout process. Only after this testing is complete should users be allowed access to the upgraded live applications.

6. Plan Creation

All required activities previously mentioned should be estimated and entered into a Project Plan in which you can identify milestone dates.

The process and considerations involved when upgrading are the subject of the rest of this document and are not covered in detail in this appendix. However, when creating a plan, you should include those elements relevant to the particular environment, at least as bullet-point tasks, if not in detail.

You may want to include the following headline stages and activities in your plan:

- **Initiation**
  - Document current environments, applications, and software versions.
  - Capture business requirements and expectations.
  - Identify key personnel and roles.
  - Define the upgrade approach.
  - Define the testing strategy.
  - Define the rollout/backout approach.
  - Produce a first draft of your Upgrade Plan. (Like any plan this will require continual refinement as the project progresses and you gain more understanding.)

- **Development and Test Environment Upgrading**
  - Back up the environment.
  - Install OpenROAD 2006.
  - Rebuild the applications.
  - Resolve any incompatibility issues encountered.
- **System Testing**
  - Produce the system test plan.
  - Execute the system test.
  - Resolve any issues encountered.
  - Obtain sign-off.
- **User Acceptance Testing**
  - Produce a user acceptance test plan.
  - Execute the user acceptance test.
  - Resolve any issues encountered.
  - Obtain sign-off.
- **Rollout to Live**
  - Back up the environment.
  - Install OpenROAD 2006.
  - Rebuild the applications.
  - Verify and test the upgraded applications.
  - Make the decision to go live or back out.
- **Project Closure**

INGRES CORPORATION : 500 ARGUELLO STREET : REDWOOD CITY, CALIFORNIA 94063
PHONE +1.650.587.5500 : FAX +1.650.587.0000 : www.ingres.com : For more information, contact info@ingres.com