amdocsbillingplatform

Amdocs Billing Platform 6.0

Upgrade Guide (Runtime)

Document Information

Software Version:	**6.0**
Publication Date:	**January 2005**
Catalog Number:	**212288**

Contents

1. INTRODUCTION

The Amdocs Billing platform provides telecommunication companies with an easily implemented Customer Care & Billing system.

The Amdocs Billing platform is comprised of software packages that run on a distributed, client-server UNIX platform. The front-end client for customer services is Web-based, while the business parameter clients are NT-based. The Amdocs components work together with a variety of third-party software and tools developed by Amdocs.

The upgrade of the Amdocs Billing platform from version 5.5 to version 6.0 requires the configuration, synchronization, and upgrade of these varied and interdependent hardware and software packages.

Amdocs has produced a comprehensive *Amdocs Billing 5.5 to 6.0 Upgrade Kit*, which provides a set of documents with step-by-step instructions, scripts and files, to enable each Amdocs account to upgrade the version 5.5 core layer and data to version 6.0. Any additional customizations made by an account's Infrastructure team should be packaged and added to this Upgrade Kit (as described in this document) and used to upgrade the customized areas before performing the upgrade.

Scope of this Guide

This document describes the steps and guidelines for upgrading the Amdocs Billing platform component testing runtime environments from version 5.5 to version 6.0.

It contains the following sections:

- Chapter 1, "Introduction" – Provides a general introduction to the document.
- Chapter 2, "Upgrade Overview" – Provides general guidelines for upgrading the system infrastructure in the component testing environments. It illustrates the main stages of the upgrade process as a high-level workflow and summarizes the activities that must be performed by the various groups working on the product.
- Chapter 3 "Upgrade Prerequisites" – Describes the system prerequisites for upgrading from version 5.5 to version 6.0.
- Chapter 4 "Preparing for the Upgrade Process" – Provides instructions for preparing the environment for the upgrade. This includes configuring the central initialization files, installing Amdocs Installation Manager (AIM) and migrating Operational jobs to dStudio-JEdi (an Amdocs-developed tool for handling jobs). It also includes instructions for adjusting the Upgrade Kit to suit your specific needs.
- Chapter 5 "Pre-Upgrade Activities" – Provides instructions for preparing the version 5.5 environment for upgrade.

- Chapter 6 “Upgrade Activities” – Provides instructions for the Infrastructure team to upgrade development and testing environments
- Chapter 7 “Implementation Guidelines” – Provides instructions for making the changes required in the implementation layer.
- Chapter 8 “Post-Upgrade Activities” – Provides instructions for all activities that must be performed on the newly upgraded version 6.0 environment.
- Chapter 9 “Testing Upgrade Guidelines” – Describes guidelines by which an account’s testing calendar should be determined.
- Appendix A, Known Validation Errors in the Product Catalog – Describes known validation errors in the Product Catalog.
- Appendix B, Focal Point Table – Provides a generic list of the various focal points involved in the upgrade.

Upgrade Approach

The following subsections provide general guidelines for upgrading Amdocs Billing platform from version 5.5 to version 6.0.

Third-Party Upgrade or Installation

Amdocs Billing platform 6.0 requires that most third-party software to be upgraded or installed. Most of the third-party software can be installed during the upgrade preparation phase.

note *Do not refer to old versions of third-party tools, as they may not be accessible.*

After a third-party tool is upgraded, the previous version of the tool may no longer be accessible (depending on the installation requirement of the specific tool).

Database Configuration

The following approaches should be considered for the database configuration:

- Upgrade the 6.0 environment in the old Oracle instances (those in which the version 5.5 environments are located)
- Upgrade the 6.0 environment using separate Oracle instances – This approach is safer if you want to secure the old version 5.5 environments.

Data Migration

In version 6.0, the generic data required by the core product is provided with the version. In addition to this generic data, each account must set up any relevant implementation data, as well as any additional attribute data required by the customization layer (if customizations have been made).

Currently, there are two sources of generic data in the product:

- IDAT files – XML files in the CC that include table structures and data. When the generated files are run, the previous core data is replaced with the new set of data. No changes are made to customized data by IDAT files and it is assumed that the core data has not been manually changed by an account in the previous version.
- Configuration applications – Data is manually updated by a component, such as the Billing Configurator, using DMP files or implementation scripts.

Each account must configure the implementation data after loading the core data. This supports the basic reference data required in the testing environment.

Upgrade Kit Contents

The *Amdocs Billing 5.5 to 6.0 Upgrade Kit* includes documents, scripts and files that enable each Amdocs account to upgrade the version 5.5 core layer to version 6.0. These documents, files and scripts must be available before starting the upgrade process.

The main process of the upgrade is to prepare and upgrade the Upgrade Kit.

Scripts and Files

The scripts and packages required for the upgrade process are provided in the Upgrade and Installation Package.

Database upgrade scripts are divided into two types:

- UDAT – Entered in dStudio-Fox for minor changes, such as upgrade values
- Application scripts for complex data upgrade changes

Since version 6.0 supports a sourceless environment, the CC is not part of the Upgrade Kit contents. Only the SDK needed for the customization layer (LEL) is provided.

Assumptions

The following assumptions apply to the *Amdocs Billing 5.5 to 6.0 Upgrade Kit*:

- The upgrade applies to the conversion of Amdocs Billing platform version 5.5 Service Pack 10 to version 6.0 Service Pack 0.

  note

  If you have a lower service pack installed on your version 5.5 environment, be sure to install Service Pack 10 before beginning the upgrade. If you have a higher service pack installed when you adjust the Upgrade Kit, additional adjustments may be necessary.

- The Upgrade Kit covers only the core modules supplied in Amdocs Billing platform version 6.0.
- Since the upgrade process is not recoverable, a rollback to the initial state may only be made using relevant backups, when required.

- Client packages are reinstalled on user PCs using the supplied installation packages and instructions.
- Oracle software with the correct version is installed on all servers involved in the upgrade (only software is required, not the instance installation).

Target Audience

This document is intended for all personnel that are involved in the planning, testing and implementation of the upgrade process. It is targeted mainly at the account's Development, Infrastructure and Testing personnel. Production upgrade is not described in this document.

Prior Knowledge

This Upgrade Guide assumes the user is familiar with Amdocs Billing platform, its UNIX and Windows components, Oracle databases, standard system administration practices, and basic knowledge of third-party tools.

Related Documents

- *Amdocs Billing Platform Server Pre-Installation Guide*
- *Amdocs Billing Platform Server Installation Guide (Runtime)*
- *Amdocs Billing Platform Client Installation Guide (Runtime)*
- *Amdocs Billing Platform Upgrade Guide (CDE)*
- *Amdocs Billing Platform Operational Guide*
- *Amdocs Installation Manager (AIM) User Guide*
- *Tiger User Guide*
- *Screen Composer 8.7.4 Guide*
- *Product Catalog User Guide*
- *dStudio - Job Editor (JEdi) 2.2 User Guide*
- *Infrastructure Release Notes*
- *Collection Implementation Guide*

Concepts and Terms

The following subsections describe the terms and acronyms that are used in this document.

Runtime Environment

A runtime environment consists of many elements, and the specific deployment of components on these elements. Some elements are private to the environment and some are shared between several or all environments. Following is a short summary of environment elements:

UNIX Accounts

- Work account – The account in which all processes operate (batch jobs, daemons, and so on). This account contains the WebLogic servers. It is also known as the user account.

  In version 6.0, the user account also contains AMC, UAMS, Back-End and Front-End.

  In a simplified environment, all the above components can be installed in one UNIX account.

- Central initialization account (shared) – The account that initializes all UNIX accounts with environment variables.

Oracle Database

Version 6.0 supports simplified environments. This means that all the areas listed below are located in a single schema in the database.

- Application account – Contains customer information.
- Reference account – Contains business parameters.
- Operational account – Contains parameter and scheduling data for batch jobs and daemons.
- Security account (shared) – Contains the data for all users in the system. This data includes user names and passwords.

TimesTen Database

Version 6.0 supports simplified environments. This means that there is one datastore for the following applications

- A&F data store (shared) – Contains data in memory required by Guiding
- Rater data store (shared) – Contains data in memory required by the Rater and Online Charging.

File Systems

- Storage (shared) – Directory contains all the executables and products required to run the software

Server

When used in this document, the term *server* refers to one of the following, depending on the context:

- WebLogic/SONAR server
- UNIX machine

AIM

The Amdocs Installation Manager (AIM) is a tool that provides the means to install the Amdocs product in a standardized, automatic fashion.

AIM is capable of handling a number of installation tasks, for various environments, product components and modules. Through its user-friendly, Web-based graphical user interface (GUI), you can configure, install and pack the required environment or module.

Tiger

Tiger is a tool for the creation and maintenance of databases. It comes in two versions: a stand-alone version and a version used within AIM (the Database I-Infra).

ADBA Repository

The Application Database Administration (ADBA) repository is the database account that contains the information used by both AIM and Tiger to create database user accounts and their content.

SONAR

SONAR is an AIM element that is used for creating and maintaining WebLogic server accounts, including WebLogic server instances.

It also helps other Java processes to create the properties files.

AMC

Application Monitoring & Control (AMC) is a generic utility that enables monitoring and manipulation of in-house applications and third-party tools. This tool also serves as the infrastructure platform for AIM and Tiger.

Operational

Operational is a table-driven tool with a GUI front-end that was developed by Amdocs to facilitate batch job scheduling, and to enable real-time monitoring of the running of batch jobs and their dependencies.

Schapi

The main operational utility to operate, monitor and control the processes.

AMC has a logical application called Process Management, which includes the Schapi GUI.

UAMS

User Account Management System, a 100% Java lightweight set of layered components extending the security mechanism of the Enterprise Java Bean (EJB) server, providing the accounting and authorization information needed for user authentication and authorization purposes. It also provides an application with generic interfaces that enable user account management.

Transaction Broker

The Transaction Broker is responsible for the exchange of information among numerous functional components of Amdocs Billing platform. Each component is responsible for a defined set of functions, and is decoupled and self-contained. The Transaction Broker handles the transfer of information between these components

APInvoker

The APInvoker is an Amdocs-developed tool that facilitates balancing the Transaction Broker's load and expedites the processing of transactions by parsing the XML and invoking EJB or local transactions.

Local Extension Layer (LEL)

Equivalent to a customization of Amdocs Billing platform.

Main Infrastructure Differences between Version 5.5 and Version 6.0

Item	Version 5.5	Version 6.0
CC	Full CC release	Only CDE release – sourceless version
Work account structure		Directory tree changed Supports core / custom areas
Oracle database version	9.2.0.3 (from SP 10, the Oracle software version is 9.2.0.5)	9.2.0.5
Database structure		Structure change (support flat account – one schema for all database components)
Unicode	Not supported	Support in version. The upgrade does not support migration to Unicode.
Accounts Receivable	Not part of version 5.5 (part of 5.6)	Released with 6.0 as part of the product
Environment variables	TLG name convention	ABP name convention.
Simplified environment	Not supported	Supported (at the UNIX level, database level, customization level)
Configuration Control	RCS	Xtra-C – Harvest
Daemons	OpDaemon worked with configuration files	Daemon Manager works with XML files
Daemon instances	According to configuration files	Generated from GN1 tables and XML templates
Job definitions and daemon connection definitions	Inserted via SQOs or DMP files, with Tiger.	Imported from the XML files in the SDK by Tiger (defined with a new tool)

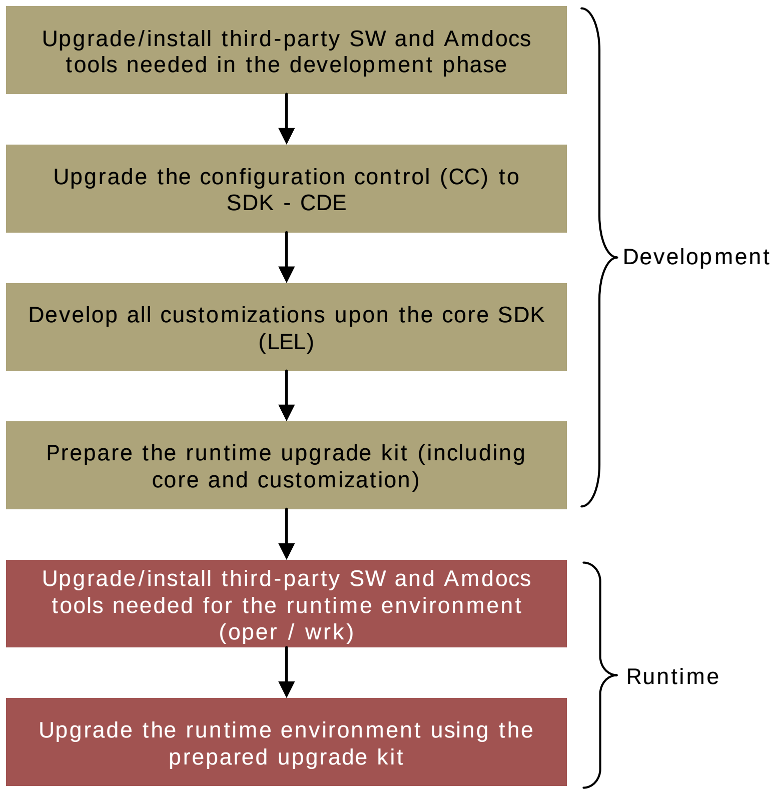
Item	Version 5.5	Version 6.0
WebLogic servers	Separate front end and back end	Unified front end and back end
AMC	Maintained the security log for users, profiles and groups in its own database schema	Maintains only the configuration repository and history table in its own schema, and uses the UAMS server to define security (users and roles)
Storage	One storage area for core and custom	Core / custom areas in the storage

2. Upgrade Overview

This chapter provides a general overview of the upgrade flow.

Upgrade Flow

The following figure describes the main steps of the upgrade flow, after the prerequisites are met:



Upgrade/install third-party SW and Amdocs tools needed in the development phase
Upgrade the configuration control (CC) to SDK - CDE
Develop all customizations upon the core SDK (LEL)
Prepare the runtime upgrade kit (including core and customization)
Upgrade/install third-party SW and Amdocs tools needed for the runtime environment (oper / wrk)
Upgrade the runtime environment using the prepared upgrade kit
Development
Runtime

Figure 2-1: Upgrade flow

Separate documents are provided to handle the first three processes (refer to “Related Documents” in Chapter 1).

- Upgrade or install third-party software and Amdocs tools needed in the development phase
- Upgrade the configuration control (CC) to SDK – CDE
- Develop all customizations upon the core SDK – LEL

note

These three steps are required only if you have an LEL layer, and need to be performed only once for the version.

This document will handle only the last three processes:

- Prepare the runtime upgrade kit (including core and customization)

  note *This step is required once and prepares the runtime upgrade in the development phase.*

- Upgrade or install third-party software and Amdocs tools needed for the runtime environment
- Upgrade the runtime environment using the prepared upgrade kit

Upgrade Roles and Responsibilities

This section describes the responsibilities of the various roles involved in the upgrade process.

- Infrastructure – The infrastructure team is responsible for:
  - Upgrading or installing Amdocs tools and third-party software
  - Creating the Upgrade Kit
  - Performing the upgrade stages for development, runtime accounts, databases, and so on
- Development – The development team is responsible for:
  - Handling customizations from the previous version that must be implemented in the new version – According to application changes described in *Amdocs Billing Platform Upgrade Guide (CDE)*.
  - Developing additional customizations for the new version
  - Informing the Infrastructure team of all changes needed to prepare the Upgrade Kit
- Implementation – The implementation team is responsible for:
  - Handling all reference changes needed to support a specific customization
  - Informing the Infrastructure team of the changes to be implemented
- Testing – The testing team is responsible for testing the upgrade process according to all information given in this document

Preparations for Adjusting the Upgrade Kit

The comprehensive *Amdocs Billing 5.5 to 6.0 Upgrade Kit* provides a set of documents with step-by-step instructions and scripts to enable each Amdocs account to upgrade the version 5.5 core layer and data to version 6.0.

The *Delivery Contents* consists of the basic Upgrade Kit, into which customization changes can be integrated. The prepared Delivery Contents scripts can be run on runtime environments to upgrade only *core to core* applications.

Each Amdocs account must create an upgrade kit that includes the core and the account's LEL (database structure changes, implementation data, additional third-party software, environment variables, and so on).

Amdocs Tiger supports the creation of both the core and customization upgrade kit (combined), based on the repository supplied with the upgrade kit.

The customized Upgrade Kit must be prepared based on existing development and customizations. It must be ready before the system test environment is upgraded.

An account may set up the customized Upgrade Kit as follows:

- Adjust the upgrade kit with scripts that include the account's customizations:
  - TimesTen scripts
  - Edit the Database data upgrade scripts
  - Create the required database patch with Tiger

These scripts are delivered for the account's core, along with instructions on how to create them to include customizations.

Pre-Upgrade Activities on the Runtime Environment

Prior to starting the upgrade process, various activities need to be performed, such as running end-of-day processes and shutting down some components.

In addition, intermediate files must be fully processed, and processes must be run until the entire set is completed. For example, all Billing processes need to be completed for a cycle, and input files need to complete all processes from A&F to the Rater.

It is recommended to complete as many of the pre-upgrade activities as possible while the system is still running in order to minimize downtime.

Upgrade Activities

The actual upgrade process involves activities such as shutting down applications, converting database structures, executing database data scripts, and removing unnecessary data. All these activities assume that there is no operational environment ready in the next version. This assumption enables various groups, such as the DBA, Infrastructure and Development teams, to perform concurrent activities. In addition, both the Oracle and TimesTen databases must be fully backed up prior to the upgrade.

Post-Upgrade Activities

In this phase, all unnecessary files and redundant software are removed. Processes needed for the normal startup of the application are run.

3. Upgrade Prerequisites

This chapter describes the prerequisites for upgrading Amdocs Billing platform component testing environments from version 5.5 to version 6.0. It provides a complete list of prerequisite software, modified hardware requirements and recommended settings for important system parameters.

Upgrade Prerequisites

This section lists the UNIX accounts, file systems, database instances and database user accounts that need to be prepared by the System UNIX team and Oracle DBA team before starting the upgrade process. You can use this information as a template for your specific upgrade.

The following diagram illustrates the upgrade prerequisites:

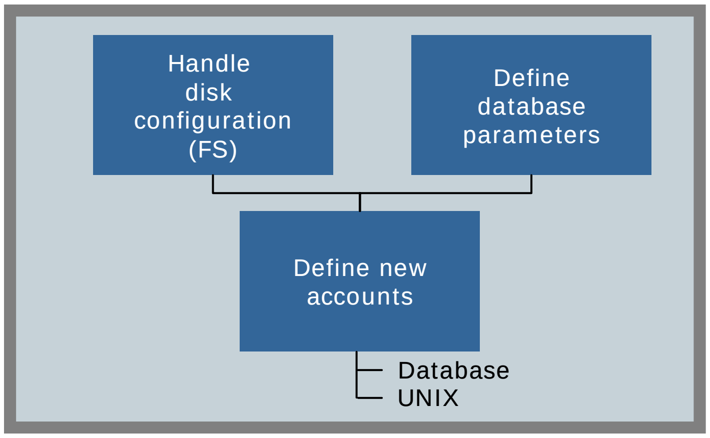


Figure 3-1: Upgrade prerequisites

Disk Configuration

For information on the required disk configuration, refer to *Amdocs Billing Platform Server Pre-Installation Guide*.

Oracle Accounts

1. Request the following accounts:

User Name	Used By	Description
AIM_DBA was created in version 5.5. Required only if Tiger will be installed in a new instance.	ADBA	Special Tiger account. Special privileges are required (see below).

User Name	Used By	Description
Oracle account for Tiger repository Required only if Tiger is installed in the new database user account.	ADBA	Tiger repository account. In version 6.0 the Oracle instance must be version 9.2.0.5. This account can be used for the AMC installation.
aimadm6	AIM	AIM AMC account.

2. Verify that you can select from the V$INSTANCE and V$SESSION tables by issuing the following SQL command:

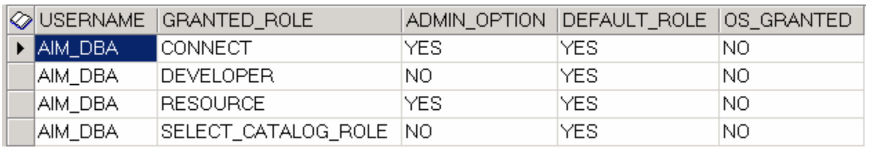
   select * from <table_name>;

3. Check that the Oracle accounts have the correct privileges:

   a. Sqlplus aim_dba/****@<instance_name>

   b. Select * from user_role_privs

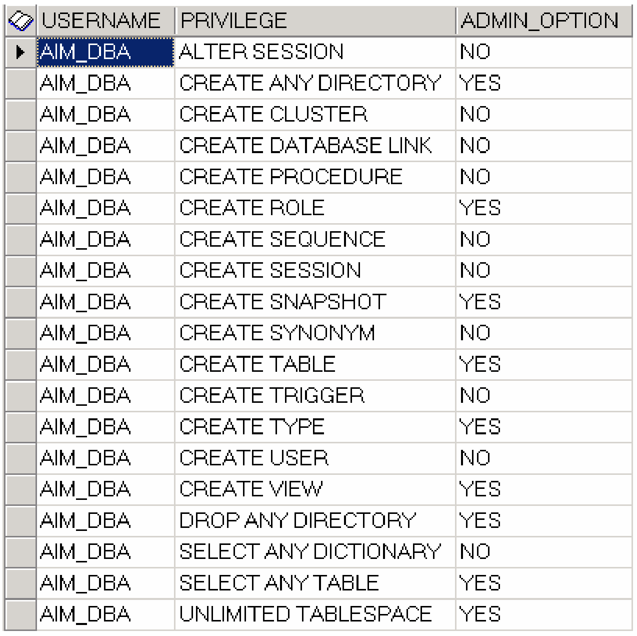
   The expected output is:



USERNAME	GRANTED_ROLE	ADMIN_OPTION	DEFAULT_ROLE	OS_GRANTED
AIM_DBA	CONNECT	YES	YES	NO
AIM_DBA	DEVELOPER	NO	YES	NO
AIM_DBA	RESOURCE	YES	YES	NO
AIM_DBA	SELECT_CATALOG_ROLE	NO	YES	NO

4. Select * from user_sys_privs

   The expected output is:



USERNAME	PRIVILEGE	ADMIN_OPTION
AIM_DBA	ALTER SESSION	NO
AIM_DBA	CREATE ANY DIRECTORY	YES
AIM_DBA	CREATE CLUSTER	NO
AIM_DBA	CREATE DATABASE LINK	NO
AIM_DBA	CREATE PROCEDURE	NO
AIM_DBA	CREATE ROLE	YES
AIM_DBA	CREATE SEQUENCE	NO
AIM_DBA	CREATE SESSION	NO
AIM_DBA	CREATE SNAPSHOT	YES
AIM_DBA	CREATE SYNONYM	NO
AIM_DBA	CREATE TABLE	YES
AIM_DBA	CREATE TRIGGER	NO
AIM_DBA	CREATE TYPE	YES
AIM_DBA	CREATE USER	NO
AIM_DBA	CREATE VIEW	YES
AIM_DBA	DROP ANY DIRECTORY	YES
AIM_DBA	SELECT ANY DICTIONARY	NO
AIM_DBA	SELECT ANY TABLE	YES
AIM_DBA	UNLIMITED TABLESPACE	YES

UNIX Accounts

1. Request the following accounts:

User Name	Used By	Group	Shell	Description
aimadm6	AIM admin account	aimsys	ksh	
dbtgr or dbtgr55 was created for version 5.5. dbtgr is required only if Tiger is installed in the new UNIX user account.	Tiger admin account	adba	ksh	
TimesTen account	TimesTen			Already exists

4. Preparing for the Upgrade Process

Preparation for the upgrade process involves performing general activities, such as verifying all third-party software versions and licenses, installing third-party software, updating scripts and environments, creating new storage, and so on. The preparation for the upgrade process is performed at different levels and depends on the current and target environments. The preparation activities are performed only once for each machine on which environments are upgraded.

The version 5.5 environment can continue to run while you install and configure a new environment for version 6.0 on the same machine, in a separate UNIX account.

It is important to start all preparations at least two days before the actual upgrade date.

The following subsections contain explicit instructions for all activities that must be performed prior to the upgrade. If you have an LEL, perform these activities only after you have upgraded the CDE (refer to *Amdocs Billing Platform Upgrade Guide (CDE)*). However, third-party software must be installed or updated first.

The following diagram illustrates the activities involved in the upgrade process.

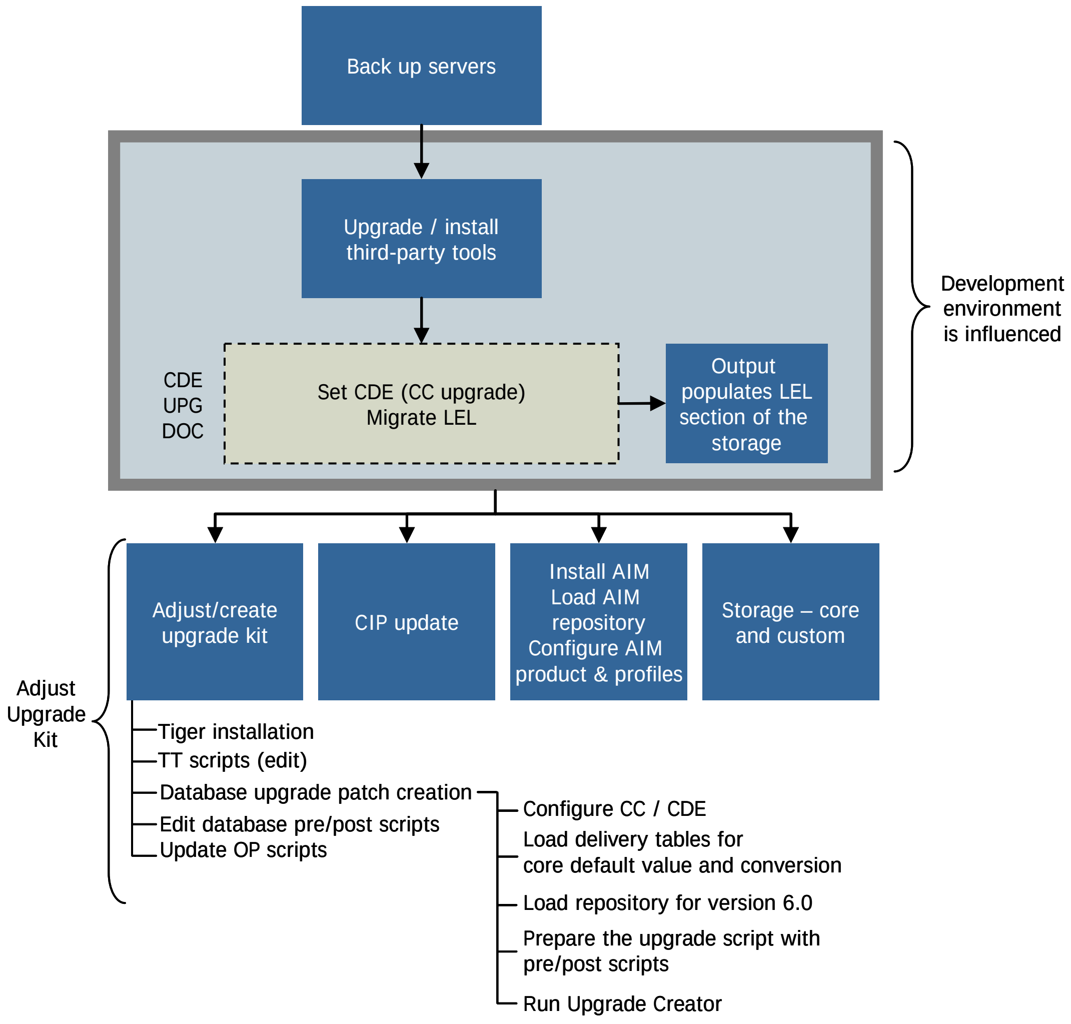

Back up servers
Upgrade / install third-party tools
CDE UPG DOC
Set CDE (CC upgrade) Migrate LEL
Output populates LEL section of the storage
Development environment is influenced
Adjust/create upgrade kit
CIP update
Install AIM Load AIM repository Configure AIM product & profiles
Storage – core and custom
Adjust Upgrade Kit
Tiger installation
TT scripts (edit)
Database upgrade patch creation
Edit database pre/post scripts
Update OP scripts
Configure CC / CDE
Load delivery tables for core default value and conversion
Load repository for version 6.0
Prepare the upgrade script with pre/post scripts
Run Upgrade Creator

Figure 4-1: Preparing for the upgrade process

Copying and Configuring the Central Initialization Account

The central initialization account includes the following files:

- .rhosts
- w.ini
- w.ini.pre

The new central initialization account for 6.0 can be configured alongside the previous central initialization account. The new configuration is made in the ~wadm/config/600 directory.

The following sections describe the changes you must make to customize the central initialization files.

To create the central initialization account:

Refer to *Amdocs Billing Platform Server Installation Guide (Runtime)*.

Configuring the w.ini.pre File

Configuring the w.ini.pre file includes both editing variables and allocating ports. When allocating ports, make sure the base port defined is identical to that of the original 5.5 environment (~wadm/config/550/w.ini.pre).The ports must be allocated from the environment's port range.

To configure the w.ini.pre file:

From the command line:

1. Change the directory to ~wadm/config/550/w.ini.pre (run cd).
2. From the version 5.5 file, make a note of the value of the first port allocated for each environment (this is usually the Usage Query port).
3. Change the directory to ~wadm/config/600/w.ini.pre (run cd).
4. Enter: vi w.ini.pre to edit the w.ini.pre file.
5. Change the default label of the work account (which came in the file template) to the name of your account.
6. Insert the values you noted into the relevant BASE_PORT sections, for each environment listed in the file.

   note

   This method of configuring ports for environment helps to streamline the upgrade process. Ports are allocated for each environment in serial order. If ports were allocated differently, this method reorders the ports for the environments. If you allocated additional ports for your customizations, they must be reallocated at the end of this process.

   For example:

```
[hbuwrk1]
BASE_PORT=14800
PORT_BASE="$(( ($ENV_NUM -1) \* 50 + $BASE_PORT ))"
subprofile env600

[hbuwrk2]
BASE_PORT=14850
PORT_BASE="$(( ($ENV_NUM -1) \* 50 + $BASE_PORT ))"
subprofile env600
```

7. Edit the following variables as illustrated in the example below (values are case-sensitive):
   - _PREFIX_L
   - _PREFIX_U
   - _PREFIX_TT

- _HOST
- _OP_ORA_USER
- _OP_ORA_PASS
- _OP_ORA_INST
- _OP_ORA_INST_NODE
- _OP_ORA_HOST
- _WL_ACCOUNT_NAME

For example:

```
[DEFAULT]
_PREFIX_L=hbu
_PREFIX_U=HBU
_PREFIX_TT=Hbu
_HOST=hpb001
_OP_ORA_USER=${_PREFIX_U}DB${ENV_NUM}
_OP_ORA_PASS=${_PREFIX_U}DB${ENV_NUM}
_OP_ORA_INST=UPG9I
_OP_ORA_INST_NODE=HPB001
_OP_ORA_HOST=${_OP_ORA_INST_NODE}
_WL_ACCOUNT_NAME=${_PREFIX_L}wrk${ENV_NUM}
```

8. Make sure the [aimadm6] section is defined.

Checking the wprofile File

To check the wprofile file:

From the command line:

1. Change the directory to ~wadm/config/600 (run cd).
2. At the command prompt, enter: vi wprofile to edit the wprofile file.
3. In the file, make sure all the references to the correct directory for the central initialization account.

Installing AIM

For instructions on installing AIM, refer to *Amdocs Billing Platform Server Installation Guide (Runtime)*. That document provides instructions for:

- Installing AIM (includes prerequisites)
- Defining AIM I-Profiles
- Configuring AIM I-Products
- Customizing AIM I-Profiles
- Configuring the Environment Monitor (EM)
- Creating an Environment Page

However, the configuration of the UNIX_BASE_PROF I-Profile is described in this document. This I-Profile must be configured as detailed in this document (refer to “Configuring I-Profiles” in this chapter).

Prerequisites

The following prerequisites must be met before you install AIM.

- Allocate a port for AIM. This port must not be in use in other areas, and must be different from that of 5.5.
- Be sure to use the aimadm6 account for the upgrade.

Configuring I-Profiles

Configure I-Profiles in the following order:

1. SONAR
   - For simplified testing environments, configure the ABP-FULL I-Profile
   - For production-like environments, configure the ABP-BEFE and ABP-UamsServer I-Profiles
2. Warehouse (storage and JavaStorage)
3. UNIX

note

There are no user-defined profiles for the Database I-Infra. Therefore, no changes are required.

Configuring UNIX I-Profile – UNIX_BASE_PROF

The UNIX I-Infra contains information for the installation of the work account. This is the account that runs the jobs and AMC. It is the main account of the environment.

The UNIX_BASE_PROF profile is used for the creation of the work environment UNIX account, including creation of the var area and initialization of the Operational and Security databases.

To configure the UNIX_BASE_PROF profile:

From AIM:

1. On the AIM Options screen, select **Aim Profile Tree**.
2. Expand the Navigation Tree to the **UNIX** node.
3. Continue expanding the tree to select **UNIX_BASE_PROF**.
4. Add the following items to the profile:

Item Type	Item Name	Execution Order
Unix_Command	Backup_Var_Config	#3
Unix_Command	Delete_TLG_HOME_Link	#4

5. To add a new item to the UNIX I-Profile:
   a. From the Type drop-down list, select the type of item to be added.
   b. In the Item Name field, add the name of the new item.

   c. Click **Add**.
   d. Use the Up and Down arrow keys to determine the order in which all items will be handled.

6. Click **Save**.

Customizing UNIX I-Profile

1. On the AIM Options screen, select **Aim Product Tree**.
2. Expand the Navigation Tree until you arrive at the UNIX_BASE_PROF I-Profile.
3. Do one of the following to access the item list for the I-Profile:
   a. In the Item column, click the button for the relevant account.
   b. Select the checkbox of the relevant account and click **Customize**.
   c. Select **All** to customize all accounts with the same values and click **Customize**.
4. In the Item Status list, click the name of the account item to be customized.
5. Modify the item according to the following table (be sure to set the order of items correctly):

Order	Item Name	Command Name	Arguments
1.	.profile	.profile	~wadm/config/600/wprofile
2.	.rhosts	.rhosts	/<new file system> /wadm/config/600/.rhosts
3.	Backup_Var_Config	tar	-cvf $HOME/var_config.Backup.tar $HOME/var/config/
4.	Delete_TLG_HOME_Link	rm	tlg_home
5.	Delete_Old_Sonar	rm -rf	$HOME/sonar/RunServer
6.	abp_home		Not customized on profile level
7.	ATL1_CreVarDir_sh	${ABP_BIN}/bin/ATL1_CreVarDir_sh	
8.	ATL1_CreWRKAccount_sh	${ABP_BIN}/bin/ ATL1_CreWRK Account_sh	
9.	ATL1_AddMarket_sh	${ABP_BIN}/bin/ ATL1_AddMarket_sh	-m m3g
10.	ATL1_MrgGnConnectParam	${ABP_BIN}/bin/ ATL1_MrgGn ConnectParam	-secuser $OP_ORA_USER -secpass $OP_ORA_PASS -secinst $OP_ORA_INST
11.	ATL1_UpdGnTaskConnect	${ABP_BIN}/bin/ ATL1_UpdGn TaskConnect	-user $OP_ORA_USER -pass $OP_ORA_PASS -mas
12.	ATL1_MngOphost	${ABP_BIN}/bin/ ATL1_MngOphost	-local

Order	Item Name	Command Name	Arguments
13.	ATL1_FillLogicalDate	${ABP_BIN}/bin/ ATL1_FillLogical Date	-del yes -ins yes -appuser $OP_ORA_USER -apppass $OP_ORA_PASS -appinst $OP_ORA_INST
14.	ATL1_OpRunSchConfig	${ABP_BIN}/bin/ ATL1_OpRun SchConfig	
15.	ATL1_AddModule_sh	${ABP_BIN}/bin/ ATL1_Add Module_sh	-m m3g -f $ABP_BIN -l 'ac amc ar art atl att aut bl cc ch cl cm de gd gn inf mf mi o2c olc op pc pm pv qmi rpl rpr sec sm tls trb tux urm rm utl vm uh up et'
16.	ATL1_FUO_Permissions	${ABP_BIN}/bin/ ATL1_FUO_ Permissions	
17.	Copy-jobmanrc	cp	-f ${ABP_BIN}/jobmanrc ${HOME}/.jobmanrc
18.	amc_batch_installer	~aimadm6/Amc-$HOST/bin/AmcBatchInstaller_amcv600.ksh	-amc_tar_gz ~aimadm6/Amc-$HOST/bin/amcv600_noBin_HP.tar.gz -sql_tar_gz ~aimadm6/Amc-$HOST/bin/amcSQL600.tar.gz -amc_db_user hbuopr1 -amc_db_pass hbuopr1 -amc_db_inst UPG9I -amc_db_host $HOST -amc_db_port 1521 -amc_port $ABP_AMC_PORT -uams_resource res/acm/client -sec_conn (Note: this is needed for configuring AMC with a Security Server UAMS; otherwise you would enter res/gen/<file_name>) $ABP_SEC_SRV_CONN
19.	update_amc_files_master	~aimadm6/Amc-$HOST/bin/AimInstallAmc_v600.ksh	-o MASTER

Order	Item Name	Command Name	Arguments
20.	update_amc_files_env	~aimadm6/Amc-$HOST/bin/AimInstallAmc_v600.ksh	-o WORK -m $USER -r $ABP_MRO_PORT -p 'PlugInSecurity Interfaces-FTG OnlineCharging Billing EventProcessing TransactionBroker OP-Maestro AR RPL ReloadReference'
21.	ATL1_InstallSchapi	${ABP_BIN}/ATL1_InstallSchapi	-i

6. Click **Apply**.
7. Click **Save Custom**.

note

If you change the Environment Code or Environment Index, this will change the screens in comparison to this document.

Migrating Jobs

At this stage, it is necessary to migrate custom jobs. This must be done in order to enable job dependencies between core and custom jobs.

Since the core development does not take custom dependencies into account, the custom development must define core parent jobs for custom jobs as required, during migration to JEdi.

Migrating Jobs to Use Table of Tables

If you have defined custom jobs that use physical parameter tables, you must migrate them to use the Table of Tables (OP_APP_DESC, OP_APP_DATA).

To migrate custom jobs to the Table of Tables:

From the command line:

- In the runtime account, run the following scripts:
    - OpConvertAppTabStruct
    - OpConvertAppTabData

Adjusting the Upgrade Kit

The *Amdocs Billing 5.5 to 6.0 Upgrade Kit* provides a set of documents scripts, and files that enable each Amdocs account to upgrade the version 5.5 core layer and data to version 6.0. Any additional changes made during development by an account’s Infrastructure team should be packaged and added to this Upgrade Kit and used to upgrade the customized areas before performing the runtime upgrade.

This section provides instructions for ensuring any individual customizations made in the version 5.5 environment or the version 6.0 CDE are maintained during the runtime upgrade process.

The information in this section is only relevant if the Amdocs Billing platform core product has been customized by an account. This section should be performed after the instructions in *Amdocs Billing Platform Upgrade Guide (CDE)* have been implemented.

If no customizations have been performed, then only the prerequisites described in “Opening the Database Delivery Contents” in this chapter must be met. The scripts provided can be used with minor changes.

Installing Third-Party Tools for the Runtime Environment

Install the required third-party and GNU software, using the configurations described in *Amdocs Billing Platform Server Pre-Installation Guide.*

C and Java compilers can be installed in different directories, but there are also patches installed in specific directories. If the upgrade is performed on the previous database instance, only one version of FastUnload can be configured to work with one database instance. This means that the new FastUnload version can only be configured after the system is shut down.

Important: If you have created the upgrade of the CDE, do not perform the steps in the rest of this chapter. Skip to Chapter 6, “Upgrade Activities”.

Opening the Database Delivery Contents

To open the database Delivery Contents:

From the command line:

1. Log in to the dbtgr Tiger UNIX account.
2. Copy the Tiger_Installation_v600.tar.gz file from the delivery contents into the $HOME directory.
3. Untar the content delivery file:

   ```
   tar –xvf Tiger_Installation_v600.tar.gz
   ```

The following directories are created under the home directory:

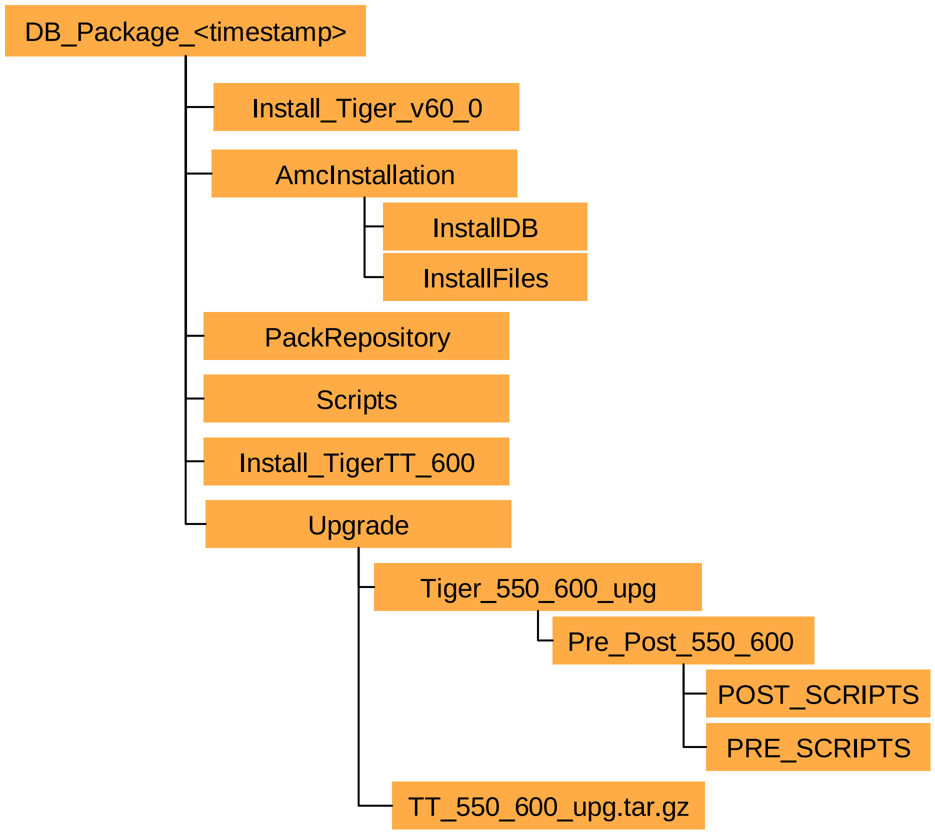


Figure 4-2: Database directory structure for Delivery Contents

To open the TimesTen Delivery Contents:

From the command line:

1. Log in to the timesten Tiger UNIX account.
2. Copy the TT_550_600_upg.tar.gz from the Delivery Contents (DB_Package_<timestamp> -> Upgrade) into the $HOME directory.
3. Extract the TimesTen Delivery Contents file:

   tar –xvf TT_550_600_upg.tar.gz

The following directories are created under home directory:

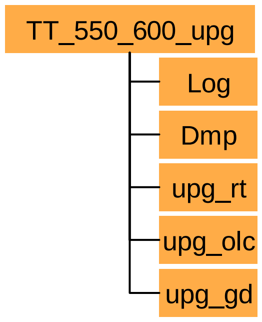


Figure 4-3: TimesTen database directory structure for Delivery Contents

Upgrading Tiger 5.5 to Tiger 6.0

Prior to upgrading Tiger to version 6.0, you must upgrade AMC to version 6.0.

If you have a CDE, Tiger must be installed only on the CDE machine. If you do not have a CDE, Tiger must be installed in the runtime environment.

Upgrading AMC to version 6.0

To upgrade AMC to version 6.0:

From the command line:

1. Log in to the Tiger dbtgr UNIX account.
2. Save the AMC directory as a backup (the directory is Amc-{host} in the home of the Tiger UNIX account.
3. Edit the following environments variables in the .profile file:

```
export ORACLE_SID=                # Apply the Oracle
instance name

#   (e.g. ORACLE_SID=AIMINST9I)
export ORACLE_HOME=           # Apply Oracle
installation home directory

#   (e.g. ORACLE_HOME=/oravl01/oracle/9.2.0.5)
export JAVA_HOME=                 # Apply Java
installation home directory

#   (e.g. JAVA_HOME=/usr/java1.4.2_06)
export NLS_LANG=              # Apply DB carachter
set
                       # (e.g.
NLS_LANG=AMERICAN_AMERICA.AL32UTF8)

# Set up Installation root directory
export installdir=$HOME/ DB_Package_<timestamp>
```

4. Run **Amc Stop**.
5. Change the directory to $install_dir/AmcInstallation/InstallDB

   cd $install_dir/AmcInstallation/InstallDB
6. Using SQL, connect to the Tiger repository user account. For example:

   <tiger_username>/<tiger_password>@<db_instance>
7. Run:

   drop table amc_conf_repository ;

   drop table amc_history_table ;

8. Run:

   @create_all.sql

   This creates five database tables used by AMC:

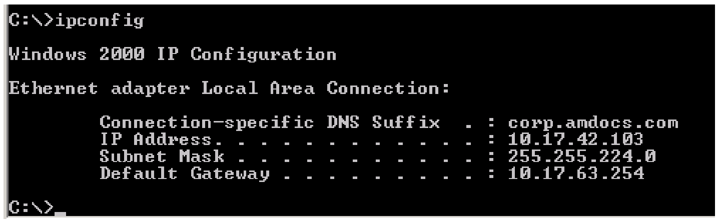
   - AMC_CONF_REPOSITORY
   - AMC_HISTORY_TABLE
   - AMC1_API_EVENTS
   - AMC1_API_TRAN
   - AMC1_API_TRAN_CELLS

9. Define the DISPLAY environment variable with your display IP address:

   export DISPLAY=<IP_Address>

   where <IP_Address> is the IP address of your computer.

   note *You can find your computer's IP address by running the ipconfig command in the Windows Command window. For example:*



```
C:\>ipconfig

Windows 2000 IP Configuration

Ethernet adapter Local Area Connection:

        Connection-specific DNS Suffix  . : corp.amdocs.com
        IP Address. . . . . . . . . . . . : 10.17.42.103
        Subnet Mask . . . . . . . . . . . : 255.255.224.0
        Default Gateway . . . . . . . . . : 10.17.63.254

C:\>_
```

   This is needed for displaying the AMC graphical user interface (GUI).

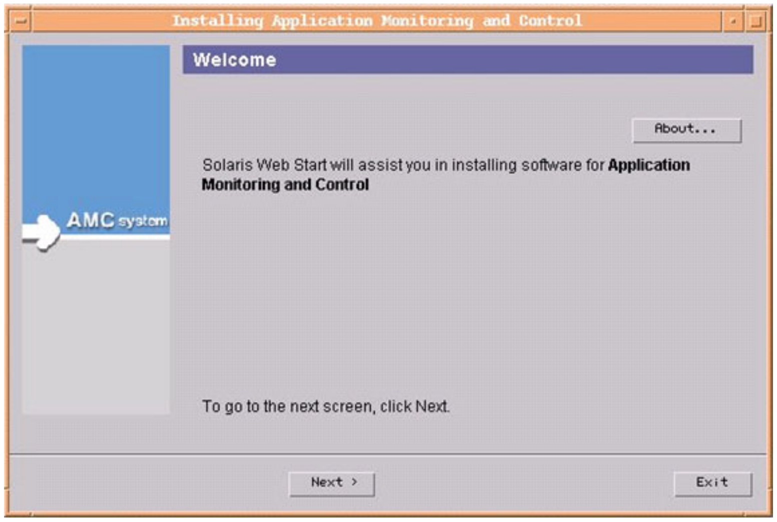
10. Change the directory to $install_dir/AmcInstallation/InstallFiles

    cd ../ InstallFiles

11. Run:

    AmcInstaller.ksh

    and wait for the GUI to open.



Solaris Web Start will assist you in installing software for **Application Monitoring and Control**

To go to the next screen, click Next.

12. Click **Next**.

13. On the Setting AMC Environment Variables screen, define the following parameters:

    This step includes selecting the port to be used by Tiger for access from your browser. Be sure to note the port you allocate.

    - Host – Server on which AMC will be installed.
    - DB Host – Host on which the AMC database account is located.
    - User Name – Amdocs Billing platform installation database user name. For a Tiger installation, enter <Tiger Account Name>.
    - Password – Amdocs Billing platform installation database password. For a Tiger installation, enter <Tiger Account Name>.
    - Instance – Amdocs Billing platform installation instance.
    - Listener port – Database listener port.

    Normally the default is sufficient.

    - AMC port – Use the port allocated for AMC in v5.5
14. Select the Custom type of installation and click **Next**.
15. On the Select Type of Install screen, click **Next**.
16. On the Locale Selection screen, click **Next**.
17. On the Select Install Directory screen, modify the installation directory:

    From: <tiger user full path>/Aim_Install/Amc-LocalHost

    To: <tiger user full path>/Amc-LocalHost
18. Click **Yes** to confirm the creation of the new installation directory.
19. On the Installation Summary screen, click **Next**.
20. On the Ready to Install screen, click **Install Now**. This action summarizes the installation data.
21. In the Installation Mode Selection screen, select the Master installation, and click **Next**.

    The next screen displays the status of the installation being carried out.
22. On the Post Installation Panel screen, click **Next** when the button becomes active.
23. To view the installation report, click **Details**.
24. Click **Exit.**
25. Change the directory to the Home directory:

    cd $HOME
26. Run .profile to initialize AMC definitions:

    ../.profile
27. Start AMC by entering the command:

    Amc Start

28. From Internet Explorer, log in to the AMC GUI using the relevant URL:

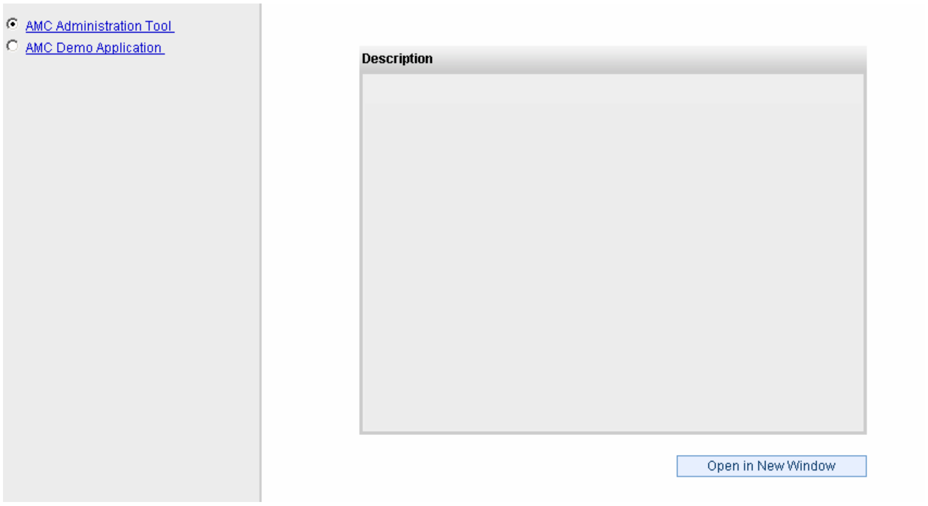
    http://<host>:<amc_port>

29. For initial login, there are some built-in users and passwords:

User	Password	
amcAdmin	Admin11	Admin user

30. To verify installation:

    a. Login as amcAdmin. The following screen is displayed:



    b. Click **AMC Administration Tool** and verify you can begin working.

31. To restore the Tiger users of the previous installation, run the AmcUamsDBConverter from Tiger UNIX ~dbtgr/Amc{host}/bin as follows:

    AmcUamsDBConverter amcAdmin/Admin11

Upgrading Tiger to version 6.0

To upgrade Tiger to version 6.0:

From the command line:

1. Change the directory to InstallTiger_v60_0 (run cd).
2. Run the InstallTiger command with the following flags (define as relevant):
    - -option – Select option 3, Upgrade Tiger Files and Upgrade Tiger Repository
    - -amcdir – AMC installation directory
    - -amcdb – Database for AMC control tables
    - -tigerdb – Database for Tiger control tables

note

The same database user account and schema can be used for both the AMC and Tiger database control tables.

- -datatbs – Tiger database table space
- -ixtbs – Tiger database index space

For example:

```
/InstallTiger -option 3 -amcdir
/dlhuser.p711/inf/aimsys/dlhtgr6/Amc-hpp711/  -tigerdb
tiger_rep/tiger_rep@STHH9I -datatbs POOL_DATA -ixtbs
POOL_IX
```

Creating a New Tiger Installation

Use this procedure if you want to create a new Tiger installation for the upgrade process instead of upgrading the existing Tiger installation.

To create a new Tiger installation:

From the command line:

1. Create a new Oracle account in the 9.2.0.5 repository.
2. Use Oracle's export/import utility to copy Tiger tables from the version 5.5 Tiger repository.
3. Drop AMC tables and sequences (all objects that start with AMC_%)
4. Create a new Tiger UNIX account called dbtgr600 and log in to it.
5. Install AMC 6.0 (refer to *Amdocs Billing Platform Server Installation Guide (Runtime)*).
6. Change the directory to $install_dir/InstallTiger_v60_0

   cd $install_dir/InstallTiger_v60_0
7. Run:

   ./InstallTiger –option 3 –amcdir <../Amc-host>
   – tigerdb <tgrusr/tgrpass@tgrinst> –datatbs <tablespace_name> –ixtbs <indexspace_name>

   where:

   - option 3 – Upgrade all Tiger files (install if they do not exist) and upgrade the Tiger repository
   - -amc dir – The full path of the AMC for Tiger installation directory.
   - -tigerdb – Database for Tiger control tables.
   - -datatbs – Data tablespace. For example, POOL_DATA
   - -ixtbs – Index tablespace. For example, POOL_IX

   For example:

   ./InstallTiger -option 3 -amcdir /dlhuser.p711/inf/aimsys/dlhtgr6/Amc-hpp711/ -tigerdb tiger_rep/tiger_rep@STHH9I -datatbs POOL_DATA -ixtbs POOL_IX

Adjusting Tiger Profiles for Version 6.0

The following subsections provide instructions for adjusting Tiger's environment profiles for version 6.0.

Defining a Simplified Environment Profile

Version 6.0 supports a simplified database structure. This means that all areas (app, ref, opr, and sec) are located in a single database schema.

Two scripts have been supplied to implement this change. These scripts insert data into the following Tiger repository tables:

- DBCONFIG_SKL
- DBRELATION_SKL
- ENV_PROFILES_SKL
- TIGER_PROFILES_TYPES

The insert automatically defines a new environment profile for Tiger, called test_global_connect. This profile contains:

- One owner containing all database objects
- One connect account containing synonyms and full permissions (select, insert, update, and delete) on the owner objects

To move from the old database structure to the new structure:

From the command line:

1. Modify the scripts to adjust them to your account's naming convention.
2. Change the directory (run cd) to:

   $install_dir/Scripts/
3. Using SQLPLUS, connect to the Tiger repository user account:

   <tiger_username>/<tiger_password>@<db_instance>
4. Run the following script:

   @test_global_connect.sql – To define the simplified environment profile with one owner and one connect.

Adding New Streams (Areas) to the Tiger Repository

In version 6.0, four new streams (areas) have been added:

- WL – Wireline
- NA – North America
- APPLCL – For Audit & Control objects that must be created in every database instance
- APPENG – For Audit & Control objects that must be created in one database instance

The new streams need only be added to complex and production environment structures, since the simplified environment profile already includes the new stream (area) definitions.

If you are not moving to the simplified profile, or you are upgrading a production environment, you must update the profiles existing in the Tiger repository with the new stream definitions (see below).

To add new stream definitions to the CCOBJAREA table:

From the command line:

1. Change the directory (run cd) to:

   $install_dir/Scripts/

2. Using SQLPLUS, connect to the Tiger repository user account:

   <tiger_username>/<tiger_password>@<db_instance>

3. Run @Add_Abp_New_Areas.sql to define the new areas.

To add new stream definitions to the existing Tiger profiles:

From the command line:

1. Change the directory (run cd) to:

   $install_dir/Scripts

2. Using SQLPLUS, connect to the Tiger repository user account:

   <tiger_username>/<tiger_password>@<db_instance>

3. To add the new area definitions to the production profile, run the following script:

   @Add_for_Prod.sql

   This script inserts data into the following Tiger repository tables:

   - DBCONFIG_SKL
   - DBRELATION_SKL
   - ENV_PROFILES_SKL

Preparing the Database Schema Upgrade

Tiger uses the pack/unpack command line utility to place the required data into Tiger repository tables. After this utility is run, the Tiger repository includes all the modules of the packed version, and the most recent CC GDD definitions of the core layer.

To unpack version 6.0 (core only):

From the command line:

1. Change the directory (run cd) to the Tiger home directory.
2. Create a temp directory under the Tiger account:

   mkdir Temp

3. Run PackTigerDbRep as follows:

   PackTigerDbRep –tigerdb <tiger_username>/<tiger_password>@<db_instance> –tempdir <the created temp dir> –unpack $install_dir/PackRepository/<name of supplied packing file>

   Be sure to use full path for the –unpack parameter.

For example:

```
PackTigerDbRep -tigerdb tiger_dls/tiger_dls@stss9i -tempdir
/tgruser.v894/dls/tgr/dlstgr/Temp/ -unpack
$install_dir/PackRepository/PACK_TIGER_600_ABSS9IU_ABP600_P
ATCH0_INITIAL_201204_114702.tar.gz
```

Configuring Tiger for Runtime Environments

The Unpack process loads the CC home definition of the core layer into the Tiger repository.

Since the runtime environment does not include a CC, *none* of the following actions need to be performed:

- There is no need to update the CC home directory.
- There is no need to load GDD files into Tiger or create a database patch.

The runtime environment uses the database objects that were supplied by the core.

To prepare the upgrade kit:

From Tiger:

The upgrade script preparation process compares the two CC states that were loaded, and collects all the information regarding default values and conversion scripts from patches run during the previous version.

1. Expand the navigation tree to the Upgrade branch.
2. Under Upgrade, click **Prepare**.
3. Select the Database Upgrade Scripts Creator tab.
4. In the Source CC area, set the following:
   - Source CC – Select **ACM550**
   - Source Patch – Select the required patch
5. In the Target CC area, set the following:
   - Target CC – Select **ABP600**
   - Target Patch – Select the required patch
6. In the Prepare Options area, set the following:
   - Select Option – Select **Populate Upg Tables**
   - Select Overwrite Upg Tables
7. Click **Prepare**.
8. To view the database changes, select the DB Reports tab and expand the relevant directory. For example, PREPARE_ABP600_20040815_154037, where the suffix is the time stamp.

Preparing the Prescripts and Postscripts

To run the prescripts and postscripts as part of the upgrade and Sync Data processes:

From the command line:

1. Copy the upgrade prescripts and postscripts from the Delivery Contents to the Tiger UNIX account, under:

   Upgrade/Tiger_550_600_upg/Pre_Post_550_600
2. Edit the prescript and postscript with the correct PATH_TO_FILES – Set the export PATH_TO_FILES with the relevant machine details.

From Tiger:

1. Expand the navigation tree from the Processes node to the Pre/Post Scripts node.
2. In the Processes list, click **Upgrade Runner**.
3. In the Pre/Post Scripts area, set the following:
   - Run Mode – From the drop-down list, select **Pre** to run a prescript.
   - Script Name Full Path – Enter the required script:

     Prescript – <full_path>/ Pre_Post_550_600/_
     Pre_Script_550_600.ksh
4. Click **Add**.
5. Click **Save Changes**.
6. The postscript must run after SyncData, and not as a post-upgrade script.

   note *For further details regarding the SyncData option, refer to "Upgrading the Environment Data (SyncData) and Running Post Upgrade Scripts" in Chapter 6.*
7. Expand the navigation tree from the Processes node to the Pre/Post Scripts node.
8. In the Processes list, click **Sync Data**.
9. In the Pre/Post Scripts area, set the following:
   - Run Mode – From the drop-down list, select **Post** to run the postscript.
   - Script Name Full Path – Enter the required script:

     Postscript – <full_path>/ Pre_Post_550_600/_
     Post_Script_550_600.ksh
10. Click **Add**.
11. Click **Save Changes**.

Identifying Renamed Objects, Columns and Other Manual Changes

Updating the UPG_OBJECTS Table

To maintain tables that were dropped in version 6.0, but are necessary for the upgrade process, the action in Tiger's upg_objects table must be changed from DROP to RENAME NO CONST.

Since Tiger does not recognize such cases, the ADBA must carefully review the DB Report to identify these changes.

The Pre_Update_UPG_OBJECTS.sql script (located under $install_dir/Upgrade/Tiger_550_600_upg/Pre_Post_550_600) handles these actions for core objects by updating the upg_objects table with the RENAME NO CONST action, old_object_name and new object_name. The script also manipulates the data for other tables that required manual changes.

Refer to the scripts for comments describing the reason to manipulate each table.

Customized objects that were dropped but that are required for the upgrade must be added to the script for the final upgrade.

The RENAME NO CONST action drops all table constraints and indexes and renames the table.

During the upgrade process, all renamed tables receive the _UPG_55_60 suffix. For example, TABLE1 is renamed to TABLE1_UPG_55_60.

To run the script and update the Tiger repository:

From the command line:

1. Change the directory (run cd) to the $install_dir/Upgrade/Tiger_550_600_upg/Pre_Post_550_600
2. Using SQLPLUS, connect to the Tiger repository user account:

   <tiger_username>/<tiger_password>@<db_instance>
3. Run the @Pre_Update_UPG_OBJECTS.sql

Updating the UPG_COLUMNS Table

Columns that were renamed between version 5.5 and 6.0, before Tiger and Fox supported the Rename Column option, must be handled manually in the upg_columns table. The Pre_Update_UPG_COLUMN.sql script modifies the entries in upg_columns for such cases.

Refer to the script to view the list of columns that were renamed, and the related tables.

To run the script and update the Tiger repository:

1. Change the directory to the $install_dir/Upgrade/Tiger_550_600_upg/Pre_Post_550_600
2. Using SQLPLUS, connect to the Tiger repository user account:

   <tiger_username>/<tiger_password>@<db_instance>
3. Run the @Pre_Update_UPG_COLUMNS.sql script.

Handling Dynamic Partitions in the Tiger 5.5 Repository

In version 5.5, dynamic partitions were not supported by Tiger.

To avoid the need to rebuild dynamically partitioned tables, the partition details must be updated in the Tiger repository.

To update the Tiger repository with details of dynamic partitions:

1. Change the directory to:

   $install_dir/Upgrade/Tiger_550_600_upg/Pre_Post_550_600

2. Using Sqlplus, connect to the Tiger repository user account:

   <tiger_username>/<tiger_password>@<db_instance>

3. Run @PreUpdatePartitionDetails.sql, which creates the following:
   - Package – update_partition_details_pg
   - Tables
     - part_vals_55_60
     - part_vals1_55_60

4. Using SQLPLUS, run the update_partition_details_pg.upd_part_dets('U') package to update the Tiger repository as follows:

   exec update_partition_details_pg.upd_part_dets('U')

   The package updates the following tables:
   - CCOBJECTS
   - PARTITION_VALUES

Running Upgrade Creator

Run Upgrade Creator after all information required for the upgrade is updated in the UPG tables.

To run Upgrade Creator:

From Tiger:

1. Expand the navigation tree from the Upgrade node to the Create node.
2. Select the Database Upgrade Scripts Creator tab.
3. From the Target CC dropdown list, select **ABP 600**.
4. Click **Run Upgrade Creator**.

note

After verifying that the Upgrade Creator has finished successfully, truncate the TIGER_UPG_DELIVERY table. This ensures the table does not affect patches in the future.

Saving Version 5.5 Customized Screens

During the upgrade process, Tiger runs a series of IDAT files to create certain core screens. These IDATs are supplied in the Delivery Contents. When the IDAT files are run, all existing screens are overwritten.

Before the upgrade:

In Screen Composer:

1. Create a Master database in which to integrate the core data with data for additional screens created in version 5.5.
2. Save data regarding any additional screens before the upgrade.

5. PRE-UPGRADE ACTIVITIES

Prior to starting the upgrade process, various activities need to be performed on the version 5.5 environment to ensure that the end-of-day processes are run and that the applications are shut down in the correct order. In addition, a full back up of both the Oracle and TimesTen databases needs to be made.

It is recommended to complete the pre-upgrade activities at least eight hours before the start of the upgrade process.

It is also recommended that processing of any intermediate files is completed, and that all processes in a set of processes are run until the entire set is completed. For example, all Billing processes need to be completed for a cycle, and input files need to complete all event processing processes from A&F to the Rater.

It is recommended to update reference tables only after the upgrade process is complete.

The following diagram illustrates these pre-upgrade activities:

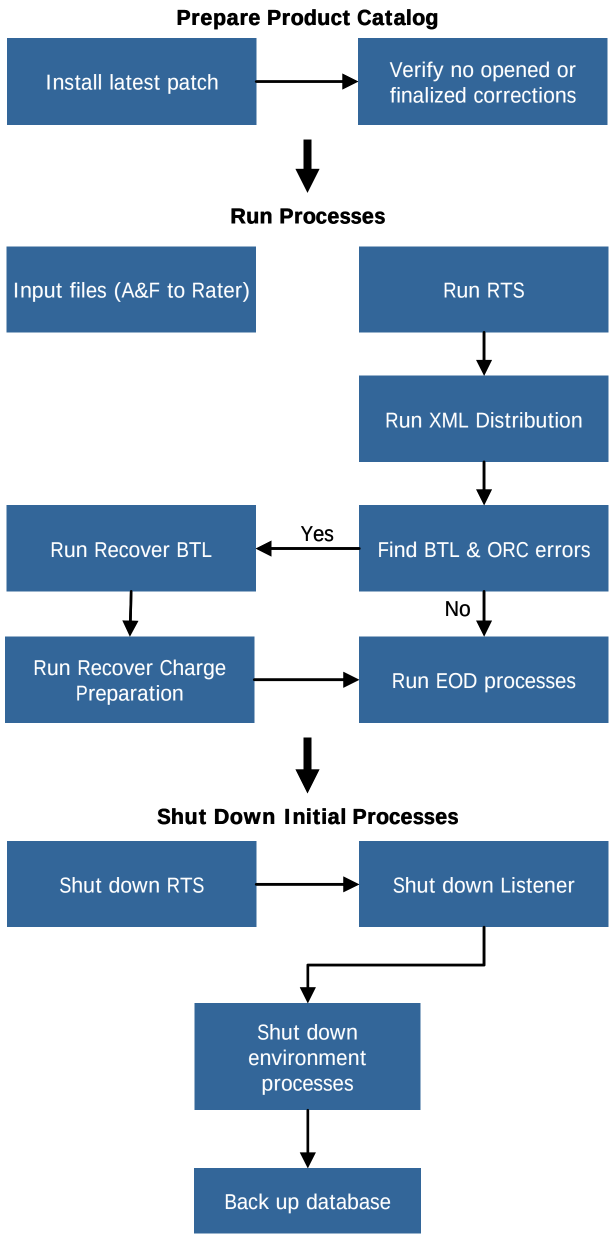


Figure 5-1: Pre-upgrade activity flow

Preparing the Product Catalog for Upgrade

This activity is performed on the version 5.5 environment.

To prepare the Product Catalog for upgrade:

From Product Catalog:

1. Verify that no open or finalized corrections exist in the Product Catalog.
2. Any opened corrections must be completed and released prior to the upgrade.

3. Verify that the latest Product Catalog patch is installed for version 5.5.
4. If the patch is not already installed, install it and run the Converter.
5. Validate Product Catalog 5.5 conversion.
   a. After conversion of the latest Product Catalog 5.5 patch, validate your model using the Global Validate option.
   b. Fix any validation errors and release the correction.

   note *Refer to Product Catalog User Guide.*

Running the Reference Table Synchronizer

The Reference Table Synchronizer (RTS) synchronizes reference table data among all application reference tables.

This activity is performed on the version 5.5 environment.

To run the RTS:

From Screen Composer:

1. From the Screen Navigator, select Utilities (UTL).
2. Under Utilities, double-click **Reference Table Synchronize (UTL1RTS)**.
3. Run **UTL1RTS**.
4. Check the results:
   - Verify that the status of all entries in the UTL1_RTS_CHANGES table is COMPLETE.
   - Verify that the RTS process copied the data from the UTL1_XML_DISTRIB reference table to the destination reference tables defined in the UTL1_RTS_SYC_RULES reference table.

Running the XML Distribution Process

Following each new release of Product Catalog versions, the XML Distribution process updates Customer Management reference tables in order to update any changes made to offers.

This activity is performed on the version 5.5 environment.

To run the XML Distribution process:

From Screen Composer:

1. From the Screen Navigator, select **CM**.
2. Double-click **CM Product Catalog Loading (CM1XMLOAD)**.
3. Run CM1XMLOAD with the appropriate parameter (RM mode).

4. Check the results:
   a. Verify that the following Customer Management tables are populated with the new offer parameters:
      - CSM_OFFER
      - CSM_OFFER_PARAM
      - CSM_OFFER_ITEM
   b. Verify that the RELEASE_STATUS for all records in the CM1_XML_DISTRIB tables is USED.

Cleaning BTL Transactions and Non-Usage Charge Errors

The Billing Transaction Listener (BTL) collects transactions for customer structures and activities that were published to the Transaction Broker from various applications, such as Customer Management, in order to update the Billing customer structure.

This activity is performed on the version 5.5 environment.

To clean BTL transactions and non-usage charge errors:

From the command line:

1. Check the following tables for error records:
   - BL1_CHG_TRX_CONTROL
   - BL1_TRX_LISTENER_ERRS
   - BL1_CYCLE_ERRORS
   - TRB1_SUB_ERRS
2. If error records are found, they must be marked and the recovery process described below must be run. If no error records are found, the recovery process can be skipped.

To run the Recover BTL and Recover Charge Preparation processes:

From AMC:

1. Select the Billing logical application.
2. Expand the Billing navigation tree from Ongoing Operations through Management.
3. Under Management, click **Billing Rejected**.
4. Verify that error records were deleted from the BL1_TRX_LISTENER_ERRS table and that records were added, updated and deleted from the following tables :
   - BL1_BLNG_ARRANGEMENT
   - BL1_CUSTOMER

- BL1_PAY_CHANNEL
- BL1_PAYER_CUST_REL (for distribution actions)
- Verify that error records are collected from the BL1_CHG_TRX_CNTRL table and an input file for the Prepare Non-Usage Rater job is prepared.

Running the End-Of-Day Map

For complete instructions on running the End-of-Day map, refer to *Amdocs Billing Platform Operational Guide*.

- Run the EOD map.

Shutting Down Environment Processes

This section describes the procedures for shutting down applications in the 5.5 environment. The upgrade calendar must be designed to ensure that the shutdown is performed in the correct order (refer to Chapter 9, “Testing Upgrade Guidelines”).

This document refers to the upgrade of a testing environment. In production environments, some of the activities can be performed simultaneously.

It is essential that processes are shut down in the order described in the table below, according to the Step column.

Step	Component / Module / Utility	Process	Pre-Activity	Procedure	Post-Activity	Remarks
1.	A&F	Listener	None	*From AMC:* Event Processing > A&F > Monitor & Control > LSN > Stop	None	
2.	Reference Table Synchronizer (RTS)			*From Screen Composer:* Run UTL1RTSHTDN.	In the UTL_RTS_CONTROL table, verify that the status of the RTS is DONE.	
3.	Online Charging	Formatting & Routing		*From AMC:* Online Charging > OLC > Monitor & Control > FR > Stop		Repeat for all Formatting & Routing instances.
4.	Online Charging	Charging Engine		*From AMC:* Online Charging > OLC > Monitor & Control > RB > Stop		Repeat for all Charging Engine instances.
5.	Online applications	Customer Management client	None	*Command line*: Run cd ${WEBLOGIC_ACCOUNT}/RunServer/config/*-FE/*Client/scripts;./stopServer	None	It is assumed that the version 5.5 server is WebLogic.

Step	Component / Module / Utility	Process	Pre-Activity	Procedure	Post-Activity	Remarks
6.	Online applications	Web server	None	*Command line*: Run cd ${WEBLOGIC_ACCOUNT}/RunServer/config/*-BE/*Server/scripts;./stopServer	None	It is assumed that the version 5.5 server is WebLogic.
7.	Guiding, Incremental Update	Update Handler Incremental Unload	Verify that there are no entries in the TRB1_SUB_LOG or TRB1_SUB_ERRS tables with a SUB_APPL_ID of 3004 or 3005.	*From AMC:* Event Processing > A&F > Monitor & Control > UHUNLD > Stop	Verify that no extract files are pending in the AC_CONTROL_01 table.	
8.	Guiding, Incremental Update	Update Handler Incremental Load	None	*From AMC:* Event Processing > A&F > Monitor & Control > UHLOAD > Stop	None	
9.	Rater, Incremental Update	Update Handler Incremental Unload	None	*From AMC:* Event Processing > Rater > Administration > Monitor & Control > CMExtract > Stop	Verify that no extract files are pending in the AC_CONTROL_01 table.	
10.	Rater, Incremental Update	Update Handler Incremental Load	None	*From AMC:* Event Processing > Rater > Administration > Monitor & Control > CM2RaterLoad > Stop	None	Repeat for all cycles.
11.	Transaction Broker		Refer to “Shutting Down the Transaction Broker – Pre-Activities” in this chapter.	*From AMC:* Transaction Broker > TRB Manager > Monitor & Control > TRBManager > Stop	None	Shut down the Transaction Broker in normal mode.

Step	Component / Module / Utility	Process	Pre-Activity	Procedure	Post-Activity	Remarks
12.	Transaction Broker	APInvoker	None	*From AMC:* Transaction Broker > TRB Manager > Monitor & Control > APInvoker > Stop		
13.	A&F	Main Driver	Check that all relevant files in the AC_CONTROL_01 table have been processed by the Main Driver.	*From AMC:* Event Processing > A&F > Monitor & Control > MD > Stop	Verify that all other A&F processes have stopped, including: ■ A&F Recycle ■ A&F Outcollect Prepare ■ SDR Exchange Rate Update ■ File To Database ■ Database To File ■ A&F Reports ■ Duplicate Check Cleanup	Repeat for all Main Driver instances.
14.	Rater	Rater	Verify that the Rater has finished processing all files from A&F.	*From AMC:* Event Processing > Rater > Rating > Monitor & Control > Rater > Stop	None	Repeat for all Rater instances.
15.	XLA	Postpaid XLA	Verify that there is no more usage in TimesTen to be replicated.	*From AMC:* Event Processing > Rater > Monitor & Control > XLAReplicator > Stop	None	Use when the account processes only postpaid events. Shut down for all DSNs.
16.	XLA	Prepaid XLA	None	*From AMC:* Online Charging > OLC > Monitor & Control > XLA_OLC > Stop	None	Use when the account processes postpaid and prepaid events.
17.	Rating	Dispatcher	None	*From AMC:* Event Processing > Rater > Monitor & Control > Dispatcher > Stop	None	Shut down for all DSNs.

Step	Component / Module / Utility	Process	Pre-Activity	Procedure	Post-Activity	Remarks
18.	Billing	Transaction Listener	None	*From AMC:* Billing > Ongoing Operations > Monitor & Control > BLBTL > Stop	None	
19.	Billing	Cycle Listener	None	*From AMC:* Billing > Cycle Operations > Monitor & Control > BLCYLS > Stop	None	
20.	Billing	Rate Change Process	None	*From AMC:* Billing > Ongoing Operations > Monitor & Control > ORCRCA > Stop	None	
21.	Billing	Prepare NU Rater Process	None	*From AMC:* Billing > Ongoing Operations > Monitor & Control > ORCPNU > Stop	None	
22.	Billing	NU Rater Process	None	*From AMC:* Billing > Ongoing Operations > Monitor & Control > pm1NURater > Stop	None	
23.	Billing	Charge Analyzer Process	None	*From AMC:* Billing > Ongoing Operations > Monitor & Control > ORCCAH > Stop	None	
24.	Billing	Load Commerce Charges Process	Check that no additional files are waiting for this process, and that the Event Processing dispatcher is down (so that no additional files are created).	*From AMC:* Billing > Ongoing Operations > Monitor & Control > COMWTDB > Stop	None	

Step	Component / Module / Utility	Process	Pre-Activity	Procedure	Post-Activity	Remarks
25.	Replenishment Management	RPL Listener	None	*From command line:* cd $ABP_BIN Enter: rpl1stop_daemon_Sh RPL1TRBSUB	None	
26.	Environment processes		None		None	■ Verify there are no processes running in either the EJB or the WRK account for the 5.5 environment. ■ Shut down any processes that are running.

Shutting Down the Transaction Broker – Pre-Activities

Complete the following activities before shutting down the Transaction Broker:

1. Shut down all the publishing applications. This includes stopping the publishing of transactions to the JMS server (such as Clarify).
2. Allow the engine to complete pulling pending transactions for distribution and verify that all TRB1_PUB_LOG tables are empty.
3. In AMC, monitor the internal queue status to verify that all transactions were published to the subscriber, and that no transactions are waiting inside the Transaction Broker engine.
4. Allow APInvoker to process all pending transactions and verify that the TRB1_SUB_LOG table for APInvoker is empty.
5. Make sure APInvoker has published all the required acknowledgments to the Transaction Broker and that the Transaction Broker has processed them.
6. Allow the subscribers to pull all pending transactions (acknowledge transactions).
7. At this stage, the following conditions should apply:
   - All transactions in the TRB1_SUB_ERR should be handled

     note *It is highly recommended that the TRB1_SUB_ERR table is also empty.*
   - The BL1_TRANSACTION_ERRS table should be empty.

6. Upgrade Activities

The actual upgrade process involves activities such as converting database structures, executing database data scripts, and removing unnecessary data. Various groups, such as the DBA, Infrastructure and Development teams, perform activities concurrently with the upgrade.

The activities described below are performed while applications are shut down.

The following diagram illustrates the activities involved in the upgrade process. Subsequent sections provide more details about each of these stages.

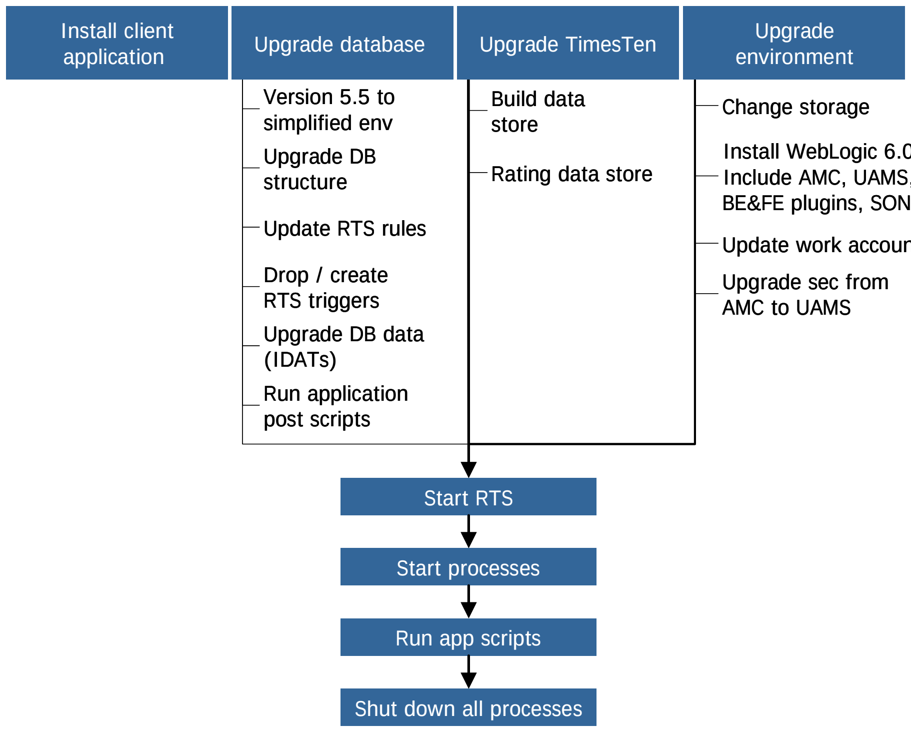


Figure 6-1: Upgrade activities workflow

Before beginning the upgrade, verify that all the applicative processes have been stopped.

Backing Up the Server and the Database

This activity is performed manually. It is not dependent upon the completion of any other activity.

- Make a complete backup of the server and database.

Installing Third-Party Software

For a complete list of required third-party software, refer to *Amdocs Billing Platform Server Pre-Installation Guide*.

Upgrading the Oracle Database Structure

The following subsections provide instructions for upgrading the structure of the Oracle database. The environment to be upgraded must exist in the Tiger repository for version 5.5.

In version 5.5, each test environment was related to a complex (security account) that could be shared by several environments.

The upgrade is performed on all environments that share the same complex. It is recommended to start the upgrade with the environment that contains the security user accounts (usually environment number 0).

Removing QMI and SM Modules from a Specific Environment

The QMI and SM modules have been removed in version 6.0.

Before upgrading the database, you must remove both modules from the environment, and close all open entries in the db_environment_details table.

To remove the QMI and SM modules:

From Tiger:

1. Expand the navigation tree from the Environment node to the Admin node.
2. Select the **Remove Module** tab.
3. On the Select Environment screen, select the required environment from the Environment drop-down list and click **Next**.

   note *If a backup is not required, clear the Backup Environment option. Select the Remove Jobs and Daemons option as necessary.*

4. On the Select Module screen, select the **QMI** and **SM** modules and click **Next**.
5. On the next screen, click **Remove**.
6. Check the log files located in:

   Tiger UNIX user – TIGER/Logs /Env/Admin/RemoveModule

Adding New Streams (Areas) to Non-Simplified Environments

This section is relevant only for non-simplified environment profiles, since the simplified environment profile includes the new area definitions.

You must add the areas to the environment before running the upgrade.

To add new streams (areas) to complex and production environments:

From Tiger:

1. Expand the navigation tree from the Environment node to the Admin node.
2. Select the **Add Area** tab.

3. On the Select Environment screen, do the following:
   a. From the Environment drop-down list, select the required environment.
   b. Clear the selection of the **Run Idat Files** option.
   c. Clear the selection of the **Run Job and Daemon Files** option.

   Since the upgrade supports the RTS process, it is very important to run the Add Area process without running the IDATs and jobs. They are run later, with the SyncData option.

   d. Leave all other defaults they are.
   e. Click **Next**.
4. On the Area Selection screen, select the new areas and click **Next** twice.
5. On the Customize Connection Parameters screen, click **Run**.
6. Check the log files located in:
   - Tiger UNIX user – TIGER/Logs /Env/Admin/AddArea
   - Tiger UNIX user – TIGER/DbBuild

Renaming the Old Environment Code

Since the same Tiger repository is used for both the old and new environments, the old env_code must be renamed to env_code_55. This enables creating the new simplified environment with the same env_code as the old one.

This section is not relevant for environments that are not moving to version 6.0's new simplified structure.

To rename the old environment code:

The following example illustrates renaming ST1 to ST1_55:

1. Use SQLPLUS to connect to the Tiger repository user account:

   <tiger_username>/<tiger_password>@<db_instance>
2. Run the following commands:
   - alter table db_environment_details disable constraint db_environment_details_1fk;
   - alter table db_environment_details disable constraint db_environment_details_2fk;
   - update db_environment_list set env_code = 'ST1_55' where env_code='ST1';
   - update dbconfig set env_code = 'ST1_55' where env_code='ST1';
   - update db_environment_details set env_code = 'ST1_55' where env_code='ST1';
   - alter table db_environment_details enable constraint db_environment_details_1fk;
   - alter table db_environment_details enable constraint db_environment_details_2fk;

Building a Simplified Environment

Create a simplified environment for version 5.5 without running IDATs or jobs, since the version 5.5 data is loaded from the old environment.

To build a simplified environment:

From Tiger:

1. Expand the navigation tree from the Environment node to the **Build** node.
2. On the Modules Group Selection screen, select **CC** and click **Next**.
3. On the CC Selection screen, click **ACM 550**.
4. On the Select Modules screen, clear the selection of the **QMI** and **SM** modules, which were removed in version 6.0.
5. Click **Next** (leave all other modules as they are).
6. On the Environment Selection screen, select the following:
   a. Select Environment Type area – Select **Test**.
   b. Select Environment Profile area – Select **TEST_GLOBAL_CONNECT**.
7. Click **Next**.
8. On the Customize Global Parameters screen, enter the required environment details and click **Next**.

   For example:
   - Environment Code – ST
   - Environment Complex – ST
   - Environment Number – 1
9. On the Set Environment Configuration screen, do the following:
   a. Select **Run Idats** and **Run Jobs and Daemons**.
   b. Leave all other defaults.
   c. Click **Next**.
10. On the next Set Environment Configuration screen, verify the instance and click **Next**.
11. On the Customize Connection Parameters for Connect Code screen, click **Save Customizations**.
12. Click **Install**.
13. Check the log files located in:
    - Tiger UNIX user – TIGER/Logs/Env/Builder/
    - Tiger UNIX user – TIGER/DbBuild

    Ignore the error regarding the UTL1_RTS_CHANGES_1IX index. This index has been dropped in version 6.0. You can also ignore the error regarding OPCOUNT. No APPLICATION_ID is defined for this table, since it is not a regular IDAT.
14. Verify that the build finished successfully.

Copying Data from the Old Environment to the Simplified Environment

To copy data from the old environment to the simplified environment:

From Tiger:

1. Expand the navigation tree from the Environment node to the **Manage** node.
2. Select the **Copy From DB** tab.
3. On the Select Environment to Import screen, select the following:
   - Target Environment Details – From the drop-down list, select the environment into which the data will be imported; that is, the new simplified environment (ST1).
   - Import Options – Select the old environment, from which the data is taken (ST1_55), and click **Next**.
4. On the Import Options screen, leave all defaults as they are.

   note *You can clear the selection of the Backup option if you want, since the new environment data is not relevant. Increase buffer size if you work with large environments.*
5. Click **Next**.
6. Click **Copy**.
7. Check the log files located in:

   Tiger UNIX user – TIGER/Logs/Env/Manager/Copy/
8. Verify that the data was copied successfully.

Handling Dynamic Partitions for Environments

In the Tiger repository, update the details of dynamic partitions for the environment to be upgraded.

To update the Tiger repository with the details of the dynamic partitions:

1. Change the directory to:

   $install_dir/Upgrade/Tiger_550_600_upg/Pre_Post_550_600
2. Using SQLPLUS, connect to the Tiger repository user account:

   <tiger_username>/<tiger_password>@<db_instance>
3. Run the following package to update the Tiger repository:

   update_partition_details_pg. ins_tiger_env_vars (<env_code>,<cc_name>)

   For example:

   exec update_partition_details_pg. ins_tiger_env_vars (‘AMH1’,’ABP’)

   The package updates the following table:

   TIGER_ENV_VARIABLES

4. Run @ PreComparePartitions.sql, which creates the following:
   - package compare_partitions_pg
   - table compare_partitions
5. From SQLPLUS, verify the results by running the following package:

   compare_partitions_pg.compare_part_vars(<env_code>,<version>)

   For example:

   exec compare_partitions_pg.compare_part_vars('AMH1','600')

Upgrading the Environment Structure

Before upgrading the environment structure, make sure the pre-upgrade process is activated.

To upgrade the environment structure:

From Tiger:

1. Expand the navigation tree from the Upgrade node to the **Run** node.
2. Select the **Upgrade Runner** tab.
3. In the Environment Details area, select the environment code of the environment to be upgraded.
4. In the Upgrade Options, set the following parameters:
   - Run Mode – From the drop-down list, select **Generate pfile and execute**.
   - Pfile Name – Enter the name of the relevant parameter file.
   - Upgrade to CC – From the drop-down list, select ABP 600.
   - Clear the selection of the **IDATs** option.
   - Clear the selection of the **Run Jobs and Daemons** option.
   - Leave other defaults as they are.

   note *Since the upgrade supports the RTS process, it is very important to run the upgrade without running the IDATs, jobs and post-upgrade scripts. They are run later, with the SyncData option.*

5. Click **Next** two times and then click **Run**.
6. After the upgrade is complete, check the log files located in:
   - Tiger UNIX user account – TIGER/Logs/Upg/Runner/
   - Tiger UNIX user account – TIGER/Upg/Status
7. The pre-upgrade scripts are run before the upgrade. Check the log for each script under:

   $install_dir/Upgrade/Tiger_550_600_upg/Pre_Post_550_600/PRE_SCRIPTS

   note *You can ignore errors regarding the UTL1_RTS_CHANGES_1IX index, since it is not built in the simplified environment.*

Adding New Modules

The following modules have been added to version 6.0:

- CCS – Customer Communication Service
- CL – Collection
- MMO – Memo
- JFD – Java Foundation
- TLS – Tools

After the environment structure has been upgraded to version 6.0, the new modules must be added to the environment in order to continue with the upgrade process.

To add the new modules to the environment:

From Tiger:

1. Expand the navigation tree from the Environment node to the **Admin** node.
2. Select the **Add Module** tab.
3. On the Select Environment screen, do the following:
   a. From the Environment drop-down list, select the required environment.
   b. Clear the selection of the **Run Idat Files** option.
   c. Clear the selection of the **Run Job and Daemon Files** option.
   d. Leave other defaults as they are.

   note *Since the upgrade supports the RTS process, it is very important to run the Add Area process without running the IDATs and jobs. They are run later, with the SyncData option.*

   e. Click **Next**.
4. On the Select Modules screen, select the new modules and click **Next** twice (be sure not to select the QMI or SM modules):
   - CCS – Customer Communication Service
   - CL – Collection
   - MMO – Memo
   - JFD – Java Foundation
   - TLS – Tools
5. On the Select Instances screen, select the relevant primary instance and click **Next** twice.

   note *To install modules on multiple instances, you must repeat the above procedure for each module.*

6. Click **Run**.
7. Check the log files located in:
   - Tiger UNIX user – TIGER/Logs /Env/Admin/AddModule
   - Tiger UNIX user – TIGER/DbBuild

Updating the Database RTS Tables and Running the RTS Triggers

Updating the Database Operational Table

Before running the RTS triggers, you must check the connections for UTL1RTS in the GN1_TASK_CONNECT and GN1_CONNECT_PARAMS tables.

To update the tables, if necessary:

From the command line:

1. Using SQLPLUS, connect to the security database user account:

   <username>/<password>@<db_instance>

2. Replace the values indicated in red with the correct environment details.

```
Update GN1_CONNECT_PARAMS set CONN_PARAMS =
'USER=STREF01;PASSWORD=ST1PASS;INSTANCE= db_instance;
PORT=1521; HOST=machine;'
Where CONNECT_CODE = (select CONNECT_CODE from
GN1_TASK_CONNECT where TASK_NAME = 'UTL1RTS' and
SESSION_ARG = 'DEFAULT' and
RUN_MODE='F' );
```

Running RTS Triggers

This activity *must* be performed before you can upgrade the data. However, it *cannot* be performed until AIM has installed all required accounts. Therefore, instructions for this activity are given in “Updating the RTS Tables” at the end of this chapter.

Upgrading Components

The following components *must* be upgraded before you can upgrade the data. However, they *cannot* be upgraded until AIM has installed all required accounts. Therefore, instructions for these components are given later in this chapter, as follows:

- Rating – Refer to “Setting Partitions for Rating Tables”.
- Resource Management – Refer to “Setting Resource Management Reference Data”.
- Replenishment Management – Refer to “Setting Partitions for Replenishment Management”.
- Extract Tool – Refer to “Upgrading the Extract Tool”.

Upgrading the Environment Data (SyncData) and Running Post Upgrade Scripts

You can only upgrade the data after the following activities have been successfully completed:

- New areas for version 6.0 have been added (refer to “Adding New Streams (Areas) to Non-Simplified Environments” in this chapter)
- Database structure has been upgraded (refer to “Upgrading the Oracle Database Structure” in this chapter)
- New modules for version 6.0 have been added (refer to “Adding New Modules” in this chapter)
- RTS triggers have been built (refer to “Updating the Database RTS Tables” in this chapter)
- Partitions for Rating tables have been set (refer to “Setting Partitions for Rating Tables” in this chapter)

To upgrade the environment data:

From Tiger:

1. Expand the navigation tree from the Environment node to the **Admin** node.
2. Select the **SyncData** tab.
3. In the Environment Details area, select the environment code of the environment whose data is to be synchronized.
   a. Select the **IDATs** option.
   b. Select the **Run Jobs and Daemons** option.

   note

   Tiger calls the JEdi Updater engine, which implements all Operational database changes. Tiger also sends all XML files for jobs and daemons to the JEdi Updater engine. For more information, refer to dStudio - Job Editor (JEdi) 2.2 User Guide.

   note

   Make sure that the post syncData process is activated (refer to “Preparing the Prescripts and Postscripts” in Chapter 4).

4. Click **Next** and then click **Run**.
5. After the upgrade is complete, check the log files located in:
   - Tiger UNIX user account – TIGER/Logs/Env/Admin/SyncData
   - The post upgrade scripts are run after the SyncData process. Check the log for each module under:

     $install_dir/Upgrade/Tiger_550_600_upg/Pre_Post_550_600/POST_SCRIPTS/MODULE_UPG

You can ignore the following errors:

- OPCOUNT – No APPLICATION_ID is defined for this table, since it is not a regular IDAT.
- INSERT INTO URM1_REF_SEQUENCES unique constraint (URM1_REF_SEQUENCES_PK) violated – Since APPLICATION_ID for version 5.5 IDATs was updated before the upgrade process, and data from version 5.5 was not deleted from this table.

note *Be sure to run the Release-Distribute process after all data changes are done (after the database upgrade and the implementation upgrade).*

Removing Unnecessary Tables

During the upgrade Tiger creates backup tables with the _UPG600 and _UPG_55_60 suffixes. After verifying that the upgrade has finished successfully, you can drop these tables from your database.

Removing Environments from the Tiger Repository

After moving to the simplified structure, you can remove the version 5.5 environment if you want.

To remove the version 5.5 environment:

From Tiger:

1. Expand the navigation tree from the Environment node to the **Admin** node.
2. Select the **Remove Environment** tab.
3. From the drop-down list, select the environment to be removed.
4. Under Remove Options, select the following options as desired:
   - Backup Environment
   - Remove All Objects (selecting this option makes the removal process much shorter)
5. Click **Next**, and then click **Remove**.

Removing QMI and SM Modules from Tiger Repository

After you have upgraded all environments and removed the QMI and SM modules from each environment, remove both modules from the CCMODULES table for version 6.0 (if they are defined in the Tiger repository.)

To remove the QMI and SM modules from the Tiger repository:

From Tiger:

1. Expand the navigation tree from the Repository node to the **Admin** node.
2. Select the **CC** tab.
3. In CC Versions Summary, select **ABP 600** and click **Edit**.
4. Click **Next**.

5. In the CC Modules Summary, select the rows of the QMI and SM modules.
6. Click **Delete**.
7. Click **Save Changes**.

Adding Synonyms

To add synonyms:

In the connect account:

- Create synonyms for the owner accounts of the following objects, of the TYPE type:
  - GN1_TAB_ATTR_TP
  - TAB_ATTR_BUFF

  These objects reside in the app area, in the GN module. They are used by the external table procedure created for the I/O framework.

  note *While you can create synonyms manually, it is recommended that their creation is implemented through Tiger's Pre/Post mechanism.*

Upgrading TimesTen

The following subsections describe the procedures for upgrading TimesTen into a simplified environment.

Upgrading the Data Store's TimesTen Version to 5120

This upgrade is performed using the ttMigrate command.

To back up the data stores and create two dump files:

From the command line:

1. In the UNIX timesten account, change into the $install_dir /TT_550_600_upg directory (run cd) and set the environment using:

   source /opt/TimesTen/tt5028_64/demo/ttSetEnv.csh

   note *All actions connect to the data store in direct mode.*

2. Stop all connections to your data stores: both online connections and automatic connections scheduled from CRON (if any). Ensure these connections stay disconnected until the entire upgrade procedure is finished.
3. Repeat the following steps for each of your defined data stores, using ttIsql and ttMigrate:

   a. ttIsql -connStr "dsn=<Your DSN>;ExclAccess=1" -e "call ttCkptBlocking(2,3); call ttCkptBlocking(2,3); quit"

   b. ttMigrate –c <DSN> <dmpFile> <owner.%>

   note *The DSNs are defined in /var/TimesTen/sys.odbc.ini.*

For example:

- Guiding – ttMigrate –c DSN= Amhwrk1_550Maf_64S_A gd.dmp amhwrk1.%
- Rating – ttMigrate –c DSN= Amhwrk1_550Rt_64S_A rt.dmp amhwrk1.%

4. Set your server DSNs in /var/TimesTen/sys.odbc.ini to point to the new 5120data stores according to the naming convention below. In Amdocs Billing platform 6.0, all TimesTen objects are created in one data store (Guiding, Online Charging, Pricing Engine and Rater processes). Therefore, the DSN naming convention has changed to:

   <prefix><#num>_600_64S_<seq_letter>

   For example:

```
Amh1_600_64S_A
```

5. Implement this change in the /var/TimesTen/sys.odbc.ini file as in the following example:

   In /var/TimesTen/sys.odbc.ini, add the DSN according to the following example:

```
[ODBC Data Sources]
AMH1_600_64S_A=TimesTen 5.1 Driver

[AMH1_600_64S_A]
Driver=/opt/TimesTen/tt5120_64/lib/libtten.sl
DataStore=/ttvl02/TIMESTEN/AMH1_600_64_5120_A/ds/AMH1_
600_64_5120_A
LogDir=/ttvl02/TIMESTEN/AMH1_600_64_5120_A/log
PermSize=8
Authenticate=0
UID=amhwrk1
```

6. In the UNIX timesten account:

   a. Create the ds and log directories.

   b. In the UNIX timesten account, set the environment using:

      source /opt/TimesTen/tt5120_64/demo/ttSetEnv.csh

7. For each dump file, run the following command:

   ttMigrate –r <DSN> <dmpFile>

   For example:

   - GD – ttMigrate –r DSN=Amh1_600_64S_A gd.dmp
   - Rating – ttMigrate –r DSN=Amh1_600_64S_A rt.dmp

8. Connect to the new 5120 data store to verify that the TimesTen data store have been created correctly in the new version.

9. Use the following ttIsql command to connect to the data store and verify its existence:

   ttIsql –connStr dsn=<dsn>

10. After you have connected to the data store, run the tables command to display the tables:

    Command> tables;

Upgrading Guiding Objects

To upgrade Guiding objects:

From the command line:

1. Change into the upg_gd directory (run cd).
2. Run the upgrade script.

   Usage:

   GD_550_to_600 <DSN> <owner> <Log Dir> <ttsqlFilesPath>

   Parameters:

   - DSN – Connection string to the version 6.0 data store
   - Owner – Owner of the tables within the data store
   - Log Dir – Directory for log output files (built by the TAR file)
   - ttsqlFilesPath – Path to the *.ttsql files of Guiding objects (in the account's own CC, this is the location containing the updated files for Guiding)
3. Check the log files located in the log directory used when the script was run.
4. Run a full extract to populate the tables with data.

Upgrading Rating Objects

To upgrade Rating objects:

From the command line:

1. Change into the upg_rt directory (run cd).
2. Run the first upgrade script.

   Usage:

   CUS_550_to_600 <DSN> <owner> <Log Dir> <ttsqlFilesPath>

   Parameters:

   - DSN – Connection string to the version 6.0 data store
   - Owner – Owner of the tables within the data store
   - Log Dir – Directory for log output files (built by the TAR file)
   - tsqlFilesPath – Path to the *.ttsql files of the Rater (Pricing Engine) Usage objects (in the account's own CC, this is the location containing the updated files for the Rater)
3. Check the log files located in the log directory used when the script was run.

4. Run the second upgrade script.

   Usage:

   USG_550_to_600 <DSN> <owner> <Log Dir> <ttsqlFilesPath>

   Parameters:

   - DSN – Connection string to the version 6.0 data store
   - Owner – Owner of the tables within the data store
   - Log Dir – Directory for log output files (built by the TAR file)
   - tsqlFilesPath – Path to the *.ttsql files of the Rater (Pricing Engine) Usage objects (in the account's own CC, this is the location containing the updated files for the Rater)

5. Check the log files located in the log directory used when the script was run.

Upgrading Online Charging Objects

To upgrade Online Charging objects:

From the command line:

1. Change into the upg_olc directory (run cd).
2. Run the upgrade script.

   Usage:

   OLC_550_to_600 <DSN> <owner> <Log Dir> <ttsqlFilesPath>

   Parameters:

   - DSN – Connection string to the version 6.0 data store
   - Owner – Owner of the tables within the data store
   - Log Dir – Directory for log output files (built by the TAR file)
   - tsqlFilesPath – Path to the *.ttsql files of Online Charging Usage objects (in the account's own CC, this is the location containing the updated files for Online Charging)

3. Check the log files located in the log directory used when the script was run.

Defining the TimesTen Database for an Upgraded Environment in the Tiger Repository

This section provides instructions for defining the TimesTen database in the Tiger repository. This section is relevant only for accounts that use the TimesTen database. All other accounts may ignore this section.

The directory tree, Install_TigerTT_600, enables the ADBA to build TimesTen environments and create and run patches on TimesTen environments.

The scripts provided by the installation kit can be executed manually or automatically by Tiger post-process scripts. For example, building a TimesTen environment might be defined as a post-process of Tiger Environment Builder.

For more information about the automation of the process, refer to "Defining Scripts as Post-Process" in this chapter.

Installing the TimesTen Support Kit

Tiger must be installed before you can install the TimesTen Support Kit (refer to *Amdocs Billing Platform Server Installation Guide (Runtime)*.

To install the TimesTen support kit:

From the command line:

1. Change the directory to $install_dir/Install_TigerTT_600 (run cd):

   cd $install_dir/Install_TigerTT_600

2. Run the following script:

   Install_TigerTT –option 2 -tigerdb <Tiger Database>

   note *The **-option 2** switch installs the TimesTen support files and control tables, and <Tiger Database> is the full database connect string for the Tiger database control tables.*

3. To enable executing commands on remote TimesTen UNIX user accounts, add Tiger's installation details (host and user) to the .rhosts file of the timesten UNIX account of the machine on which you are installing the environments.

4. The TimesTen scripts use remsh (remote shell). In order for it to work properly, the references to prompt (with stty commands) in the UNIX account's TimesTen initialization files (.cshrc, .login) must be wrapped within *if*.

   For example:

```
if ( $?prompt ) then
stty erase "^H"
stty erase "^h" kill "^U" intr "^C" eof "^D" susp "^Z" hupcl ixon ixoff tostop
endif
```

   This refers to TimesTen UNIX accounts on the machines on which the environments are created.

5. Verify that the timesten user has read and write permissions on the /var/TimesTen/sys.odbc.ini file.

6. Verify that a TIMESTEN directory exists under /ttvl02 owned by the UNIX user timesten group dbtimes.

Defining Scripts as Post-Process

This optional step enables running TimesTen scripts automatically by Tiger as post-process scripts. Use the Tiger graphical user interface to define the following scripts:

- Builder_TT.ksh as a post-process of Tiger Environment Builder
- PatchCreator_TT.ksh as a post-process of Tiger Patch Creator
- PatchRunner_TT.ksh as a post-process of Tiger Patch Runner
- Remove_env_TT.ksh as a post-process of Tiger Remove Environment
- Checker_TT.ksh as a post-process of Tiger Checker

After the installation, the scripts are located in ~/Amc-${host}/bin.

To define a script as post process:

From Tiger:

1. Expand the Navigation Tree to the **Processes** branch (Tiger DB Tools → ADBA → Processes).
2. Under Processes, click **Pre/Post Scripts**. The Process List is displayed.
3. Perform the following steps for each of the scripts listed above:
   a. In the Process List, click the relevant Process Name. For example, to add Builder_TT.ksh, click the **Environment Builder** process. The Handle Pre/Post Scripts screen is displayed.
   b. From the Run Mode drop-down list, select **POST**.
   c. In the Script Name Full Path field, click the Browse button (...) to locate and select the script you are defining.
   d. From the Active Script Ind drop-down list, select **Y**.
   e. Click **Add**.
   f. Click **Save Changes**.
   g. Click **Pre/Post Scripts** to return to the Process List.
   h. Each added script appears under the relevant process name in the Process List screen.

Adding TimesTen Definitions to the TEST_GLOBAL_CONNECT Profile

To add TimesTen definitions to Tiger's TEST_GLOBAL_CONNECT profile:

From the command line:

1. Log in to the dbtgr Tiger UNIX account.
2. Change the directory to $install_dir/ /Scripts (run cd).

   cd $install_dir/Scripts
3. To define the simplified environment profile for TimesTen, run @test_global_connect_tt.sql.

Identifying the TimesTen Environment in the Repository

To identify the TimesTen environment in the Tiger repository:

From the command line:

1. In the dbtgr Tiger UNIX account, change the directory to:

   $HOME/Amc-<machine>/bin cd $HOME/Amc-<machine>/bin
2. Run the Define_Exist_Env.ksh script:
3. Define_Exist_Env.ksh <Tiger_Rep> <CC_Name> <CC_Ver> <Env_Code>

   For example:

   Define_Exist_Env.ksh tiger_600/###@inst ABP 600 AMH1

Upgrading Environment Accounts

Perform the following activities, in the following order, to upgrade the environment accounts:

note *The storage and WebLogic accounts are actually installed, not upgraded.*

- Install the Warehouse I-Infra
- Upgrade the UNIX I-Infra
- Install the SONAR I-Infra – ABP-FULL

Installing the Warehouse I-Infra

To install the Warehouse I-Infra:

From AIM:

1. On the AIM Options screen, select **AIM Product Tree**.
2. Expand the AIM Product Tree until you arrive at the relevant I-Product (AIM System → AIM Product Definition → Products → <I-Product>).
3. Select the **Product Installation** tab. The I-Module Selection screen is displayed.
4. Click **Clear Configuration** to clear the previous installation choices.
5. To install storage and JavaStorage in all modules, select all I-Modules and click **Next**. A list of I-Infras is displayed.
6. Select **Warehouse I-Infra**. The details of the configured storages and JavaStorages are displayed.
7. Select the relevant storage and JavaStorage, and click **Next**. The AIM Installation screen is displayed.
8. Select the following:
   - AIM Installation – Select **Install**
   - Log Mod Priority – Select **Debug**
9. Click **Next**. The Installation Options screen is displayed.

10. Select **Product**, and click **Next**. The final installation screen is displayed.
11. Click **Start Installing** to install the storage and JavaStorage.

Upgrading the UNIX Work Account

To upgrade a UNIX work account:

From AIM:

1. On the AIM Options screen, select **AIM Product Tree**.
2. Expand the AIM Product Tree navigation tree until you reach the relevant I-Product (AIM System → AIM Product Definition → Products → <I-Product>).
3. Select the **Product Installation** tab. The I-Module Selection screen is displayed.
4. Click **Clear Configuration** to clear the previous installation choices.
5. Select the Base I-Module and click **Next**. A list of I-Infras is displayed.
6. Select **UNIX I-Infra**. The details of the UNIX work accounts are displayed.
7. Select the relevant UNIX work account and click **Next**. The AIM Installation screen is displayed.
8. Select the following:
   - AIM Installation – Select **Install**
   - Log Mod Priority – Select **Debug**
9. Click **Next**. The Installation Options screen is displayed.
10. Select **Product**, and click **Next**. The final installation screen is displayed.
11. Click **Start Installing** to install the UNIX account.

Installing SONAR I-Infra – ABP-FULL

To install the SONAR I-Infra:

From AIM:

1. On the AIM Options screen, select **AIM Product Tree**.
2. Expand the AIM Product Tree until you arrive at the relevant I-Product (AIM System → AIM Product Definition → Products → <I-Product>).
3. Select the **Product Installation** tab. The I-Module Selection screen is displayed.
4. Click **Clear Configuration** to clear the previous installation choices.
5. Select the Base I-Module and click **Next**. A list of I-Infras is displayed.
6. Select **SONAR I-Infra**. The details of all the configured SONAR I-Infras are displayed.
7. Select the ABP-FULL I-Profile and click **Next**. The AIM Installation screen is displayed.

8. Select the following:
   - AIM Installation – Select **Install**
   - Log Mod Priority – Select **Debug**
9. Click **Next**. The Installation Options screen is displayed.
10. Select **Product**, and click **Next**. The final installation screen is displayed.
11. Click **Start Installing** to install all WebLogic domain accounts.

SONAR Post-Installation Activities

The following subsections describe activities to be performed after the SONAR I-Infra has been installed.

Verifying BOH Security

After the SONAR ABP-FULL I-Profile is installed, perform the following:

- Run the .profile file from the home directory:

  . ./.profile

Since this version supports the Amdocs Billing platform BOH tree, the root namespace name that was used in UAMS 0.49 ($) must be changed to the new root namespace name that is equal to the BOH root ID.

To verify BOH security:

From the command line:

1. Enter the SEC1_NAME database table and locate the row in which the NAME column value = $.
2. Manually change the value of the NAME and PATH columns to the ID of the root BOH node (usually 0).

Adjusting Environments to Storage

To adjust the environment to the new storage:

From AIM:

1. On the AIM Options screen, select **EM Tree**.
2. Expand the EM System tree to the environment (EM System → EM → <host> → <env_code>).
3. Right-click the WebLogic account of the environment being installed.
4. Select **SONAR Domain Maintenance**.
5. Right-click a J2EE domain name and select **Adjust to Storage**.
6. Click **Save Custom**.
7. Repeat for all other SONAR domains.

Updating the Operational Database

The following subsections provide instructions for updating the Operational database.

Example of Inserting Implementation Data – Optional

This section provides instructions for inserting sample data into the Operational database. These instructions are optional.

If you do not want to use the provided implementation example, you must create another implementation. Without such implementation, Online Charging, Rating and A&F will not function.

Furthermore, for environments other than simplified system test environments (such as production or distributed environments) additional implementation for Audit & Control is required.

The following components provide implementation data:

- A&F
- Online Charging
- Rating
- Audit & Control

Because the data affects the GN1_CONNECT_PARAMS table, the Security database is also affected.

To insert implementation data into the Operational database:

1. Ensure the Security database is defined with write permissions.
2. In the op area of the database, run the following scripts:
    - Acquisition & Formatting
        - ABP_UPG_MF_V550_V600_TASK_CONNECT_DATA_POST_PATCH.sql
    - Rating
        - Rater_Implementation_600_OP.sql
    - Online Charging
        - gn1_art_process_config.sql
        - gn1_art_section_param.sql
        - gn1_task_connect.sql
    - Audit & Control
        - Task_Connect.ProduciotnLike.sql

Updating Operational Data

After the upgrade is complete, some Operational tables must be updated according to the environment data. The following tables must be updated:

- For specific hosts and users:
  - OPHOST
  - OPPROG_SCH
- For database connections:
  - GN1_TASK_CONNECT
  - GN1_CONNECT_PARAMS

For a System Test environment whose connection configuration matches that of the APDO System Test (with regard to connect code standards), use the scripts below to configure these tables.

To update Operational data:

From the command line:

- From the prepared version 6.0 storage, run the following Operational scripts with the num option:
  - For specific hosts and users, run:

    ATL1_MngOphost -local

    This script updates OPHOST and OPPROG_SCH.
  - For the GN1_TASK_CONNECT table, run:

    ATL1_UpdGnTaskConnect -mas -num
  - For the GN1_CONNECT_PARAMS table, run:

    ATL1_MrgGnConnectParam -def -del yes -ins yes -num

    Check the -h flag for additional options, such as providing the security database connection if the security schema is different from the Operational schema.
  - To populate the logical date for applications, run:

    ATL1_FillLogicalDate -def

Starting the WebLogic Server

To start the WebLogic server:

From the command line:

1. Change to the $WEB_ROOT_DIR/ABPServer/scripts directory (run cd).
2. Enter the command: **./startServer**.
3. View the log files to verify that the ABP-FULL server started correctly. The log files are located in:

   $WEB_ROOT_DIR/ABPServer/logs/weblogic.XXXXXX_XXXXXX.log

Converting AMC Security

The version 5.5 environment contained configuration files that are not used in version 6.0. Therefore, after the UNIX_BASE_PROF profile is installed, you must convert the AMC user accounts to the new security mechanism.

To convert AMC user accounts:

From the command line:

1. Log into the AMC Master for the version 5.5 environment.
2. In this account, run the following command:

   echo $AMC_CONNECT_STRING
3. Make a note of the value returned by the echo command (the <connect string value>).
4. In the account being upgraded, make sure AMC is not running (run Amc Stop if necessary).
5. In the account being upgraded, run the following command:

   export AMC_CONNECT_STRING=<connect string value>
6. Change the directory to <Amc-host>/config (run cd).
7. Run the following command:

   ~aimadm6/Amc-$HOST/bin/AmcUamsDBConverter amcAdmin/Admin11

Installing Client Applications

For instructions on installing client applications, refer to *Amdocs Billing Platform Client Installation Guide (Runtime)*.

Updating the RTS Tables and Creating RTS Triggers

To update the RTS tables and create RTS triggers:

From the command line:

1. Using SQLPLUS, connect to the environment's database user account:

   <username>/<password>@<db_instance>
2. Customize the following scripts according to the needs of the account:
   - ABP_UPG_UTL_V550_V600_UTL1_RTS_SYC_RULES.sql – Populates the RTS_SYC_RULES table with version 6.0 triggers.
   - ABP_UPG_UTL_V550_V600_UTL1_RTS_CALLBACK.sql – Populates the RTS_CALLBACK table with the version 6.0 callbacks.
3. Run the scripts.

4. Create a ksh script named UTL1_setRTSUpgConnects. This script exports all the connect parameters of the database in which the triggers are created:
   - export GN1_ORA_USER=User Name
   - export GN1_ORA_PASS=Password
   - export GN1_ORA_INST=Instance

   If there are remote databases that the RTS must also synchronize (MASTER_COMP and TARGET_COMP columns in the UTL1_RTS_SYC_RULES table), export the connection parameters for those databases as follows:
   - export [SESSION_ARG]_USERUser
   - export [SESSION_ARG]_PASSWORD=Password
   - export [SESSION_ARG]_INSTANCE=Instance
5. Place UTL1_setRTSUpgConnects in the $PATH variable and change its permissions to *executable* (chmod 755 UTL1_setRTSUpgConnects).
6. Run UTL1rts_upgrade_Sh, which also runs the UTL1_setRTSUpgConnects script.
7. Check that all triggers were created successfully.

Setting Partitions for Rating Tables

The following Rating tables must be updated:

- RPR1_PARTITION_DEFINITION
- RPR1_CYCLE_PARTITION_REL
- PM1_CYCLE_STATE

To update Rating tables:

note ***You may only update the Customer Management or Billing tables after you update and adjust the Rating tables.***

1. Run the following SQL statement from the ref area of the database:

   Rater_Implementation_600_REF_BackComp.sql

   This script populates the RPR1_PARTITION_DEFINITION and RPR1_CYCLE_PARTITION_REL tables with sample data.
2. Adjust the data in the tables to suit your specific implementation:
   - Number of Rating partitions
   - Relationships between cycles and partitions
3. Adjust the following columns in the PM1_CYCLE_STATE table:
   - PREV_CYCLE_INST – Enter the ID of the previous cycle instance.
   - NEXT_CYCLE_INST – Enter the ID of the next cycle instance.
   - PARTITION_ID – By default, populated with a value of 1 when the table is updated. For each existing partition, duplicate each table row and change the value of this column accordingly.

Setting Resource Management Reference Data

In Resource Management

The following Resource Management tables are supplied with default values:

- RM_RESOURCE_TYPE
- RM_PACKAGE_TYPE
- RM_PACKAGE_TP_ATTR

note *The reference data must be updated before the post-upgrade script for Resource Management is run.*

To set Resource Management reference data:

1. Check the default values carefully.
2. Adjust them as necessary, according to the following table:

Table	Column	Description	Default value	Remark
RM_RESOURCE_TYPE	RRT_PATITION_KEY	Defines the Unified Resource Type partition Key (This value is crucial, since the first partition level is set according to this value) For more details regarding this, please refer to the data model changes section.	MOD ((2* ROWNUM) , 100);	If the core defaults for this field does not meet the specific requirements of the upgrading project, then the upgrading project *must change* this value at this stage
RM_PACKAGE_TYPE	RPT_PATITION_KEY	Defines the Unified Resource Type partition Key (This value is crucial, since the first partition level is set according to this value) For more details regarding this, please refer to the data model changes section.	MOD ((2* ROWNUM + 1) , 100)	If the core defaults for this field does not meet the specific requirements of the upgrading project, then the upgrading project *must change* this value at this stage
RM_PACKAGE_TP_ATTR	RPTA_SRCH_KEY1_IND	Defines weather the current attribute value should be maintained in the Packages' SearchKey1 field. (In order to allow quick search according to this attributes value) P.S. The core defines the 'PARAM_NAME' attribute as the value of the SearchKey1 field.	DECODE (RPTA_ATTR_NAME, 'PARAM_NAME', 'Y' , 'N')	If the core defaults for this field does not meet the specific requirements of the upgrading project, then the upgrading project *must change* this value at this stage

Setting Partitions for Replenishment Management

In version 5.5, the RPM_PROFILE table had the following entries:

- RCG_MO_PAR=5
- BKT_MO_PAR=5

1. Make a note of the values of these entries.

   note *Be sure to note these values **before** you upgrade the database.*

2. In version 6.0, update the following entries with these values:
   - RCG_MO_PAR – Replace the default value with that of version 5.5.
   - BKT_MO_PAR – Replace the default value with that of version 5.5.
   - MONTH_PAR – Update with the relevant number of months of Replenishment history you want to save for subscribers.

     note *This value must match that of version 5.5. By default, the value is 12 months.*

   - DAY_PAR – Update with the relevant number of days of Replenishment history you want to save for subscribers.

     note *This value must match that of version 5.5. By default, the value is 7 days.*

Upgrading the Extract Tool

The following subsections provide instructions for upgrading the Extract Tool.

Preparing the Extract Tool Repository for Upgrade

A post-upgrade script updates Extract Tool data automatically, during the post-database upgrade process (refer to “Upgrading the Environment Data (SyncData) and Running Post Upgrade Scripts” in this chapter). However, you must edit the script with the required customization layer code.

To prepare the Extract Tool repository for upgrade:

From the command line:

1. Change the directory to:

   $install_dir/Upgrade/Tiger_550_600_upg/Pre_Post_550_600

2. Edit the Post_Script_550_600.ksh script with the required customization layer code.

   Mandatory fields are not always populated by default with the correct value.

3. Edit one of the following scripts according to the content of the Extract Tool repository to be upgraded:
   - When the repository to be upgraded contains only *core* data, without any customization, run the following script:

     ABP_UPG_getV550_V600_update_cl_code_core.SQL
   - When the repository to be upgraded does not contain core data (that is, a core implementation), run the following script:

     ABP_UPG_getV550_V600_update_cl_code_no_core.SQL
   - When the repository to be upgraded contains both core implementation and customization implementation, run one of the following scripts:
     - ABP_UPG_getV550_V600_update_cl_code_core_and_cust.sqlPost_Script_550_600.ksh (default)
     - ABP_UPG_getV550_V600_update_cl_code_core.sql
4. Run the relevant script.

Running the Extract Tool Converter

The Extract Tool Converter must be run prior to using Extract Tool 6.0.

To convert Extract Tool data to version 6.0:

1. Ensure Extract Tool 6.0 has been installed. Since the converter is a C#-based executable, it must be run from a Windows platform with .Net 1.1 Framework and MDAC2.7 (refer to *Amdocs Billing Platform Client Installation Guide (Runtime)*).
2. In the Extract Tool $install_dir/db/converter directory, extract the contents of the ABP_UPG_getV550_V600_global_data_conversion.zip file.
3. Extract the following zip file to the folder of your choice:

   ABP_UPG_getV550_V600_global_data_conversion.zip

   The directory tree is created, which enables running the required data conversion executable
4. Configure the repository's connection prior the execution of executable:

   a. Change the directory to the xmls sub-directory where you unzipped the files.

   b. Edit the repository_info.xml file and change the content of the following tag:

```
<Backend name="repository"
connectionString="Provider=OraOLEDB.Oracle;Password=[Ente
r password here];User ID=[Enter user name here];Data
Source=[Enter data source here]" role="0" type="ole">.
```

   c. Enter the username, password and instance of the repository that needs to be upgraded.

5. Run the following executable:

   <unzipped dir>\ABP_UPG_getV550_V600_global_data.conversion.exe.

   A log file called patch.log is created in the executable's directory. The patch runs several sub-patches internally. It also creates a file for each of the sub-patches, with the following name convention:

   ETP<patch number>

   In the event of recovery, these files indicate which sub-patch must be run.

6. After the Converter has finished running, check the .\patch.log files for errors. If no error is found, the repository is ready for version 6.0.

7. To verify the repository was successfully upgraded to version 6.0:

   a. On the subsystems screen (Data Object, Extract), ensure you can see and work with your converted elements.

   b. On the Project screen, check that the projects are configured.

Running the Product Catalog Converter

The Product Catalog converter requires active RTS triggers. These triggers are activated by DBA upgrade scripts.

To convert Product Catalog data to version 6.0:

1. Ensure the Product Catalog 6.0 has been installed (refer to *Amdocs Billing Platform Client Installation Guide (Runtime)*).

2. From the Windows Explorer, double-click the *Product catalog converter.exe* file located in

   ..//Support tools/ProductCatalogConverter of the released patch version. The Product Catalog Converter dialog box is displayed.

3. Next to the Configuration File field, click the **Browse** button and select the *Product catalog config.xml* file.

4. In the DB Connection field, verify that the user account instance is the database on which the Product Catalog instance runs.

5. Click **Convert**.

   The Converter automatically recognizes the version or patch with which the data is compatible, and performs the conversion from it.

6. Check the result status of the converter execution:

   a. In the PC1_CONVERSION table, select the entry with the highest CONVERSION_DATE.

   b. Ensure that the value of the CURRENT_VERSION column is the name of the Product Catalog patch to which you are converting the data.

   c. Ensure that the value of the STATUS column is Success.

      - A log file of the conversion process is saved in the library from which you ran the executable file. It is called Converter.<date of execution>.log.

      - If the conversion failed, it is possible to re-run the converter.

7. Select database encoding – In the PC1_POLICY_VARS table, the DBEncoding variable indicates the encoding type used in distribution. Select the required encoding type.
8. After the conversion process has finished successfully, the Validate All Versions dialog box is displayed. To validate the converted implementation, click **Validate**.
9. Fix validation errors. This activity is done in Product Catalog 6.0. Refer to Appendix A, "Known Validation Errors in the Product Catalog".
   a. Finalize corrections.
   b. Release the version.
10. Distribute all versions.

7. Implementation Guidelines

This chapter provides guidelines and instructions for implementing changes resulting from the upgrade of Amdocs Billing platform.

A&F is populated according to the account's specific configuration. Therefore, it is delivered without data.

The instructions for A&F implementation are provided with example data.

Customer Care Service, Memo and Collection are new components in version 6.0.

Utilities Implementation

This section describes the changes made to Utilities Management as a result of changes to its data model:

The following changes have been made:

- Tables with dropped IDATs
  - UTL1_RTS_SYC_RULES
  - UTL1_RTS_CALLBACK

The change defines the new entries that contain new rules with new conditions specific to each application. The UTL1_RTS_SYC_RULES table's callback function is updated according to the new rules.

You need take no further action to implement Utilities.

Acquisition & Formatting Implementation

This section describes the major implementation changes made in version 5.5 for A&F reference data.

Optional changes can be performed manually if applicable.

MF1_APPLICATION

A field called DATA_GRP_PREFIX has been added to this table.

This field determines the data group prefix that is mapped by the A&F processes (Listener, Splitter, Main Driver, Append, Recycle, Outcollect Prepare) in the relevant Audit & Control table (AC1_SITES_ROUTING).

The data group value determines the routing criteria for the A&C mechanism and therefore the values populated in MF1 APPLICATION must be synchronized with the content of AC1_SITES_ROUTING.

The *Get Next* API (of Audit & Control) populates the output data group for the *Create Output* API (of Audit & Control). This is true only for the Splitter and Append processes, which work without RBMS rules.

The following table illustrates the core implementation for populating the DATA_GRP_PREFIX:

ORIGIN PROG	RECEIVE PROG	DATA_GRP_PREFIX
LSN	SPL	MF1LSNTOSPL
SPL	MD	MF1SPLTOMD
MD	APP	MF1MDTOAPP

MF1_OPERATION

The change in MF1_APPLICATION influences the following fields in MF1_OPERATION:

- ORIGIN_PROG
- RECEIVE_PROG

MF1_LOGICAL_DEST

The change in MF1_APPLICATION influences the following fields in MF1_LOGICAL_DEST:

- FUNC_TYPE
- PROCESS_NAME

MF1_DISTRIBUTE_DEST

The change in MF1_APPLICATION influences the following fields in MF1_DISTRIBUTE_DEST:

- DIST_DEST – The file type defined in MF1_APPLICATION must also populate this field.

Populating the DATA_GRP_PREFIX Field

To populate the DATA_GRP_PREFIX field:

1. Write SQL statements that draw the required data from the tables described in the subsections above.
2. In the database reference area, run the SQLs.
3. Check the tables to ensure the entries are correct.

MI1_DAYLIGHT_SAVING

The structure of the table has been changed.

The following fields have been dropped:

- YEAR
- START_MONTH
- START_DAY_OF_MONTH
- END_MONTH
- END_DAY_OF_MONTH

They have been changed to:

- FROM_DATE
- TO_DATE

To implement these changes:

1. Populate the FROM_DATE field with the start date and time of the daylight saving period.
2. Populate the TO_DATE field with the end date and time of the daylight saving period.

MF1_DISTRIBUTION_DEST

For entries of Guiding Error and Usage Events, the field DIST_DEST in MF1 DISTRIBUTE_DEST must be updated. The previous formula was:

<cycle code>_R_<cycle code><rater instance>

The new formula is:

<cycle code>_<task ID>

To update the DIST_DEST field:

1. In MF1_LOGICAL_DEST, based on the data in FUNC_TYPE, perform the Select query on KEY_ID to retrieve the KEY_ID for the Rater entries.
2. For each key that was found, go to MF1_DISTRIBUTE_DEST and manually change the DISTRIBUTE_DEST field to the new format (new format - <cycle code>_<task ID>).

MF1_APPEND_CONFIG

This section describes a change resulting from the addition of the DATA_GROUP_PROC field to the MF1_APPEND_CONFIG table. Population of this field is optional.

The field determines the data group creation procedure, which must be used for each input file type.

To populate the MF1_APPEND_CONFIG table:

- Populate the following with a value of zero (0):
    - Regular file types that are processed just as they were in previous versions
    - Creation of data groups that rely on the MF1_APPEND_CONFIG and MF1_GEN_PROPERTIES tables

note

The Append process uses the default value of zero when the field remains empty.

OUTCOL_PROVIDER

This section describes a change resulting from the addition of the PARTITION_ID and GROUP_ID fields to the OUTCOL_PROVIDER table. Population of these fields is optional.

To update the OUTCOL_PROVIDER table:

1. The *group ID* attribute can remain empty if providers do not have to be divided into logical groups for separate processing.
2. The *partition ID* attribute can remain empty if providers are not distributed between different sites (Rater partition IDs).

   If the provider is being billed, the field must be populated with the location of the outcollect event Rater cycle.

GD1_XML_CONFIG

This section describes a change resulting from the addition of the GD1_XML_CONFIG table.

To populate the GD1_XML_CONFIG table:

From the ref area of the database:

- Run the following script:

  Gd1XmlConfigLoader

Rating Implementation

This section describes the changes made to Rating as a result of changes to its data model.

Mandatory changes for version 5.5 are updated with an upgrade script. Optional changes can be performed manually if applicable.

All the changes listed in this section are mandatory changes.

The following changes were implemented by Tiger during the database upgrade (refer to Chapter 6, “Upgrade Activities”):

- New tables
  - RPR1_PARTITION_DEFINITION
  - RPR1_CYCLE_PARTITION_REL

You need take no further action to implement Rating.

Replenishment Management Implementation

This section describes the changes made to Replenishment Management as a result of changes to its data model.

Mandatory changes for version 5.5 are updated with an upgrade script. Optional changes can be performed manually if applicable.

All the changes listed in this section are mandatory changes.

The following changes have been made:

- New tables

  RPL1_OLC_MESSAGES – Required for Replenishment's workflow
- Tables with new fields

  RPM_ACTIVITY – Describes the code of a published XSD
- Tables with dropped IDATs
  - RPL_PROFILE
  - RPM_CONF_CHANNEL
  - RPM_CONF_RCG_MTD
  - RPM_VLD_CHN_TO_RCG

To update the tables:

1. Run the following SQL statements from the ref area of the database:
   - ABP_UPG_RPLV550_V600_RPM_PROFILE.sql

     note *Refer to "Setting Partitions for Replenishment Management" in Chapter 6.*
   - ABP_UPG_RPLV550_V600_RPL1_OLC_MESSAGES.sql
   - ABP_UPG_RPLV550_V600_RPM_ACTIVITY.sql

     note *This script inserts missing records regardless of the existance of unique constraints on the table's columns. Therefore, you may ignore errors resulting from unique constraints.*
   - ABP_UPG_RPLV550_V600_RPM_ACTIVITY_UPDATE_NEW_FIELDS.sql
   - ABP_UPG_RPLV550_V600_PERIOD_KEY_UPDATE.sql
   - ABP_UPG_RPLV550_V600_RPM_VLD_CHN_TO_RCG
   - _UPDATE_PARENT_KEY.sql
   - ABP_UPG_RPLV550_V600_RPL1_CYCLE_INSTANCE.sql
   - ABP_UPG_RPLV550_V600_RPM_VLD_CHN_TO_RCG.sql
   - ABP_UPG_RPLV550_V600_RPM_CONF_CHANNEL.sql
   - ABP_UPG_RPLV550_V600_RPM_CONF_RCG_MTD.sql
2. Run the following SQL statements from the app area of the database:
   - ABP_UPG_RPLV550_V600_PERIOD_KEY_UPDATE.sql

     note *This script must be activated using the database **owner** permissions; otherwise it will fail.*

Resource Management Implementation

This section describes the changes made to Resource Management as a result of changes to its data model.

Mandatory changes for version 5.5 are updated with an upgrade script. Optional changes can be performed manually if applicable.

The following changes were implemented by Tiger during the database upgrade (refer to Chapter 6, "Upgrade Activities"):

New tables:

- RM_RESOURCE_TYPE
- RM_PACKAGE_TYPE
- RM_PACKAGE_TP_ATTR

You need take no further action to implement Rating.

Customer Management Implementation

This section describes the changes made to Customer Management as a result of changes to its data model.

Mandatory changes for version 5.5 are updated with an upgrade script. Optional changes can be performed manually if applicable.

All the changes listed in this section are mandatory changes.

The following changes were implemented by Tiger during the database upgrade (refer to Chapter 6, "Upgrade Activities"):

- Tables with new fields

  ACTIVITY_GROUP_ID – Added to the CSM_ACTIVITY table. The value of this field is represented in the CM1_GENERIC_CODES tables.
- Tables with dropped IDATs
  - CSM_POLICY_VARS
  - CSM_ACTIVITY
  - CSM_ACT_RSN

You need take no further action to implement Customer Management.

Customer Management Integration with Resource Management

This section provides instructions for accounts in which Customer Management is integrated with Resource Management.

To implement this integration:

- In the CM_RESOURCE_TYPE table, populate the new field RM_IMPL_CLASS with the relevant Customer Management – Resource Management implementation class. This must be done for each resource type.

  The core-implementation class is:

  amdocs.csm3g.rm.RmCmImpl

Memo Implementation

This section describes the changes made to Collection Management as a result of the new Memo component:

New tables:

- MO1_MEMO_TYPE – In version 5.5, the MEMO_TYPE field was part of Customer Management. In version 6.0, Memo has become a component in and of itself. Therefore, in Customer Management, the data type of the MEMO_TYPE field has changed from Char(4) to Varchar2(6). The MEMO_TYPE field is the foreign key to Memo's MO1_MEMO_TYPE table. This table is populated with data from both Customer Management and Collection.
- MO1_APPLICATION
- MO1_ACTIVITY

To populate the MO1_MEMO_TYPE table:

1. Be sure the following Customer Management script has been run on the MO1_MEMO_TYPE table:
2. Be sure the following Customer Management script has been run on the CSM_ACT_RSN table:

   ABP_UPG_CM_V550_V600_CSM_ACT_RSN_POST_PATCH.sql
3. Run the following Collection script on the MO1_MEMO_TYPE table:

   ABP_UPG_CL_V550_V600_MO1_MEMO_TYPE_POST_PATCH.sql

To populate the Memo tables:

From the ref area of the database:

- Run the following SQL statements:
  - ABP_UPG_CL_V550_V600_MO1_ACTIVITY_POST_PATCH.sql
  - ABP_UPG_CL_V550_V600_MO1_APPLICATION_POST _PATCH.sql

Collection Implementation

This section describes steps you must take to enable use of Collection (a new component) after upgrading the system.

To populate the Collection tables:

From the ref area of the database:

- Run the following SQL statements:
  - ABP_UPG_CL_V550_V600_CL1_ACTIVITY_POST_PATCH.sql
    - ABP_UPG_CL_V550_V600_CL1_ACT_RSN_POST_PATCH.sql
    - ABP_UPG_CL_V550_V600_CL1_ACT_PARMS_POST _PATCH.sql
  - ABP_UPG_CL_V550_V600_CL1_AR_DEBT_TYPE_POST _PATCH.sql

- ABP_UPG_CL_V550_V600_CL1_COLLECTOR_GROUP_POST _PATCH.sql
- ABP_UPG_CL_V550_V600_CL1_ATTR_MAPPING_POST _PATCH.sql
- ABP_UPG_CL_V550_V600_CL1_FOLLOW_UP_POST_PATCH.sql
- ABP_UPG_CL_V550_V600_CL1_AGENCY_POST_PATCH.sql
- ABP_UPG_CL_V550_V600_CL1_AR_FINANCE_ACTIVITY _POST_PATCH.sql
- ABP_UPG_CL_V550_V600_CL1_CATEGORY_POST_PATCH.sql
- ABP_UPG_CL_V550_V600_CL1_XML_DISTRIB_POST_PATCH.sql
- ABP_UPG_CL_V550_V600_CL1_CUST_COMUNICAT_POST _PATCH.sql

Populating the BPA Tables

For instructions on populating the BPA tables, refer to *Collection Implementation Guide*.

Accounts Receivable, Journaling and General Ledger Implementation

This section describes the changes made to Accounts Receivable, Journaling and General Ledger Management as a result of changes to its data model:

To populate the Accounts Receivable tables:

From the ref area of the database:

- Run the following SQL statements:
  - ABP_UPG_AR_V550_V600_AR1_CHARGE_CODE_POST_ PATCH.sql
  - ABP_UPG_AR_V550_V600_AR1_CHARGE_GROUP_CODE_ POST_PATCH.sql
  - ABP_UPG_AR_V550_V600_AR1_CHARGE_GROUP_PRIORITY _POST_PATCH.sql
  - ABP_UPG_AR_V550_V600_AR1_CREDIT_REASON_POST_ PATCH.sql
  - ABP_UPG_AR_V550_V600_AR1_CUSTOMIZATION_MAPPING _POST_PATCH.sql
  - ABP_UPG_AR_V550_V600_AR1_FINANCIAL_ACTIVITIES_ POST_PATCH.sql
  - ABP_UPG_AR_V550_V600_AR1_FINC_ACTV_LOGIC_POST_ PATCH.sql
  - ABP_UPG_AR_V550_V600_AR1_GENERIC_CODES_POST_ PATCH.sql
  - ABP_UPG_AR_V550_V600_AR1_GL_DATA_COLUMNS_POST _PATCH.sql

- ABP_UPG_AR_V550_V600_AR1_GROUPING_RULE_NAMES_POST_PATCH.sql
- ABP_UPG_AR_V550_V600_AR1_INVOICE_TYPE_POST_PATCH.sql
- ABP_UPG_AR_V550_V600_AR1_PROPERTIES_POST_PATCH.sql
- ABP_UPG_AR_V550_V600_AR1_REFUND_REASON_POST_PATCH.sql
- ABP_UPG_AR_V550_V600_AR1_WRITE_OFF_REASON_POST_PATCH.sql
- ABP_UPG_AR_V550_V600_AR1_DEPOSIT_REASON_POST_PATCH.SQL
- ABP_UPG_AR_V550_V600_AR1_XML_DISTRIB_POST_PATCH.Idat

note

This script may be located in either of the following paths:
~ccip/bb/gare/v60_0/upg/src
~ccip/bb/gare/upg/src

Audit & Control Implementation

This section describes the changes made to Audit and Control Management as a result of changes to its data model:

The following changes have been made:

- Tables with dropped IDATs
  - AC_PROGRAMS
  - AC_PGM_RULES_CONTROL

To update the tables:

- Run the following SQL statements from the ref area of the database:
  - ac_pgm_rules_control_impl.sql
  - ac_programs_impl.sql

Billing Implementation

This section lists all the manual reference data updates required for core Billing data. The manual configuration may be performed only after Tiger has run the relevant upgrade scripts.

BL1_CYCLE_CODE Table

Two new fields have been added to the table columns:

- BE – Business entity
- CYC_POPULATION_CODE

These fields must be manually updated according to Customer Management data.

Billing does not support multiple business entities. Therefore, the BE column added to BL1_CYCLE_CODE is relevant for other components, but not for Billing.

This means you may not define more than one BE column in Billing.

Updating Implementation Data

This section lists all the manual updates that must be made to Billing’s reference implementation data.

AC_PGM_RULES_CONTROL

The AC_PGM_RULES_CONTROL table lists all the process relations in a specific route. The ROUTE classification did not exist in version 5.5. In version 6.0, it is represented in the FILE_FORMAT field.

- Insert all the customization records from version 5.5 with the relevant ROUTE values to FILE_FORMAT.

BL1_BILL_PROC

A new field, called FLOW_ID, has been added to the table columns.

- Manually update all non-COREDT records.

BL1_BILL_PROC_IO

The BL1_BILL_PROC_IO table has replaced the BL1_BILL_PROC_REL table. It lists all the input and output data elements for processes in a specific route.

- In BL1_BILL_PROC_IO, manually insert all records that were inserted by customization into BL1_BILL_PROC_REL with the appropriate route.

BL1_CHARGE_CODE

A new field, called CLASSIFICATION, has been added to the table columns. The core data is managed via the Reference Data Synchronizer.

- Update all non-COREDT records manually.

BL1_RC_ACT_POLICY

- Populate the BL1_RC_ACT_POLICY table manually, due to changes in version 6.0 (new fields: activity_id and activiy_obj_names).

BL1_DATA_ELEMENT

A new column, called FILE_FORMAT, has been added. This column indicates the format of a file.

- Manually update the table’s customized data.

BL1_XML_CONFIG

The following schemas have been modified:

Schema Name in Version 5.5	Schema Name in Version 6.0
cfComplexEntity	ComplexEntity
bfComplexEntity	ComplexEntity
bfExtractEntity	ExtractEntity
cfExtractEntity	ExtractEntity

The upgrade process replaces the contents of version 6.0 data only. That is, all records whose schema name is different from that listed above for version 5.5 are replaced.

The records with the above version 5.5 schema names remain in the table, to enable the migration of any customized XML files to version 6.0.

1. After the migration is complete, delete these records, since they are not used in version 6.0.
2. Manually update all new Complex Entities created in the customization, and core Complex Entities that were modified by the customization.

   The change is also made in the XML schema, so the customization user must modify the customization XML file according to the new schema.
3. Manually update the XML files for newly created Extract Entities.

Upgrading the Billing Transaction Listener

In Billing 5.5, the BL1_RATE_ALERT_ACT reference table was used by the Prepare NU Rater process to create the relevant events for the NU Rater per transaction.

In Billing 6.0, the information that existed in BL1_RATE_ALERT_ACT is defined in three different reference tables.

The table below shows the mapping between the old BL1_RATE_ALERT_ACT table and the new BL1_CUSTOMER_ACT and BL1_CUSTOMER_OFFER_ACT tables. In addition, you must define the relevant offer-related activities (for example, perform Add and Remove in the BL1_OFFER_ACT table).

Enabler 5.5		**Enabler 6.0**	
Table Name	Column Name	Table Name	Column Name
BL1_RATE_ALERT _ACT		BL1_CUSTOMER _ACT	
	ACTIVITY_NAME		ACTIVITY_ID
	ACTIVITY_ORIGIN		ACTIVITY _ORIGIN
	RC_RATE_AFFECT_IN D		RC_INDICATION
	NON_RC_RATE_IND		RC_UPDATE _DATE_IND
	OC_EVENT_IND		OC_INDICATION
		BL1_CUSTOMER_OFF ER_ACT	
	OC_OFFER _ACTIVATION_IND		SEND_OFFER _OC_IND
	OC_PENALTY_IND		SEND_OFFER _OC_IND

For example:

In version 5.5, a Service Change activity was defined in the BL1_RATE_ALERT_ACT table as follows:

- Activity name = SERVICE_CHANGE
- Activity origin = CM
- RC_RATE_AFFECT_IND = Y
- NON_RC_RATE_IND = N
- OC_EVENT_IND = Y
- OC_OFFER_ACTIVATION_IND = Y
- OC_OFFER_PENALTY_IND = Y

In version 6.0, the Service Change activity is defined as follows:

- BL1_CUSTOMER_ACT
  - CUSTOMER_ACT_SEQ = 36
  - ACTIVITY_ID = 45
  - ACTIVITY_ORIGIN = CM
  - RC_INDICATION = Y
  - RC_UPDATE_DATE_IND = N
  - OC_INDICATION = N
- BL1_OFFER_ACT
  - OFFER_ACTIVITY_ID = 100, 110
  - ACTIVITY_NAME = Added, Removed
  - SEND_WITH_RC_IND = Y, N
  - SERVICE_FILTER = OCOFFA, OCPENA
  - OC_EVENT_TYPE = OC activity event
- BL1_CUSTOMER_OFFER_ACT
  - CUSTOMER_ACT_SEQ = 36
  - OFFER_ACTIVITY_ID = 100,110
  - SEND_WITH_CA_IND = N,N
  - SEND_OFFER_OC_IND = Y,Y

Product Catalog Implementation

This section describes changes to the implementation of Product Catalog, with regard to the Flexible Bill function. While it is possible to use Amdocs Billing platform without the Flexible Bill function, these changes must be implemented.

Discount Packages

A mandatory property called *Frequency* has been added to discount packages (both regular and conditional). It is based on the *Cycle frequency* Elementary Type (the only relevant values are: None, Monthly and Weekly).

If the implementation does not include multiple frequencies, and the property was added, the value of this property must be set to Monthly (M); otherwise, the Discount Engine cannot run.

To set the Frequency property:

In the Product Catalog:

- Ensure the *Frequency* property is attached with a value of Monthly.

note *If the property is not attached at all, the system handles it as Monthly.*

Changes to External Records

To implement changes to the Allowance billing record:

1. Access the external record *Allowance billing record* (Implementation Repository → External Records → *Allowance billing record*).
2. Change the Customer ID field to Service receiver customer ID.

Changes to Recurring Charges

This section describes changes that must be made to recurring charges.

To implement changes to the RC base external record:

1. Access the external record *RC base external record* (Implementation Repository → External Records → RC base external record).
2. Add the following fields:
   - Frequency – Based on the *Cycle frequency* Elementary Type (default value = None)
   - Frequency multiplier – Numeric (default value = 1)
   - Create multiple charges indicator – Boolean (default value = No)

To implement changes to the RC base PIT:

1. Access the RC base PIT.
2. Under *Extract base attribute handler*, add a new mapping handler called *Extract frequency attribute* (Extract → Mapping RC external record QC→ Extract frequency attributes – New mapping handler).
3. Map values to the following fields:
   - *Frequency* – Populated with the value *M* (Monthly)
   - *Frequency multiplier* – Populated with the value *1*
   - *Create multiple charges indicator* – Populated with the value *No*

Payment Frequency Elementary Type

The values of the existing *Payment frequency* Elementary Type have been changed to: generation frequency 1, generation frequency 2, etc. For example, if the Frequency is set to Monthly and generation frequency = 2, a recurring charge is created every two months.

Customer Activities

In version 5.5, the Prepare NU Rater process used the Billing reference table BL1_RATE_ALERT_ACT to create relevant events for the NU Rater for each transaction. In version 6.0, the following reference tables have replaced BL1_RATE_ALERT_ACT:

- PC1_CUSTOMER_ACT
- PC1_OFFER_ACT
- PC1_CUSTOMER_OFFER_ACT

The tables are created automatically during the conversion process. After the conversion process is complete, verify that the tables were created in the reference area.

Enter the data in the above tables (General Reference Tables in the Auxiliary Repository) via Product Catalog.

The table below shows the mapping between the BL1_RATE_ALERT_ACT table and the PC1_CUSTOMER_ACT and PC1_CUSTOMER_OFFER_ACT tables.

Version 5.5		Version 6.0	
Table Name	**Column Name**	**Table Name**	**Column Name**
BL1_RATE_ALERT_ACT	ACTIVITY_NAME	PC1_CUSTOMER_ACT	ACTIVITY_ID
	ACTIVITY_ORIGIN		ACTIVITY_ORIGIN
	RC_RATE_AFFECT_IND		RC_INDICATION
	NON_RC_RATE_IND		RC_UPDATE_DATE_IND
	OC_EVENT_IND		OC_INDICATION
	OC_OFFER_ACTIVATION_IND	PC1_CUSTOMER_OFFER_ACT	SEND_OFFER_OC_IND
	OC_PENALTY_IND		SEND_OFFER_OC_IND

In the BL1_RATE_ALERT_ACT table, if either OC_OFFER_ACTIVATION_IND or OC_PENALTY_IND is set to Y, then in the PC1_CUSTOMER_OFFER_ACT table, SEND_OFFER_OC_IND must be set to Y.

In the PC1_CUSTOMER_OFFER_ACT table, SEND_OFFER_OC_IND can only be set to N if both these fields are also set to N.

After the data has been entered, RTS extracts the information from the Product Catalog tables and populates the following Billing tables:

- BL1_CUTOMER_ACT
- BL1_OFFER_ACT
- BL1_CUSTOMER_OFFER_ACT

Example of Converting Data

This section provides an example of mapping data from version 5.5 to version 6.0. Each implementation must map the tables according to the relevant business rules.

- In version 5.5, the *Service change* activity was defined in BL1_RATE_ALERT_ACT as follows:
  - Activity name = SERVICE_CHANGE
  - Activity origin = CM
  - RC_RATE_AFFECT_IND = Y
  - NON_RC_RATE_IND = N
  - OC_EVENT_IND = Y
  - OC_OFFER_ACTIVATION_IND = Y
  - OC_OFFER_PENALTY_IND = Y
- In 6.0, the *Service change* activity is defined as follows:
  - PC1_CUSTOMER_ACT

    The PC1_CUSTOMER_ACT table has one entry for SERVICE_CHANGE:
    - CUSTOMER_ACT_SEQ = 36
    - ACTIVITY_ID = 45
    - ACTIVITY_ORIGIN = CM
    - RC_INDICATION = Y
    - RC_UPDATE_DATE_IND = N
    - OC_INDICATION = Y
  - PC1_OFFER_ACT

    The PC1_OFFER_ACT table has three entries. These entries are not specific to SERVICE_CHANGE and may be used by other activities.

    Entry 1:
    - OFFER_ACTIVITY_ID = 100
    - ACTIVITY_NAME = Added with RC
    - SEND_WITH_RC_IND = Y
    - SERVICE_FILTER = OCOFFA
    - OC_EVENT_TYPE = OC activity event

    Entry 2:
    - OFFER_ACTIVITY_ID = 110
    - ACTIVITY_NAME = Removed
    - SEND_WITH_RC_IND = N
    - SERVICE_FILTER = OCPENA
    - OC_EVENT_TYPE = OC activity event

Entry 3:

- OFFER_ACTIVITY_ID = 120
- ACTIVITY_NAME = Current
- SEND_WITH_RC_IND = Y
- SERVICE_FILTER = OCCMAC
- OC_EVENT_TYPE = OC activity event

• PC1_CUSTOMER_OFFER_ACT

The PC1_CUSTOMER_OFFER_ACT table has three entries:

Entry 1:

- CUSTOMER_ACT_SEQ = 36
- OFFER_ACTIVITY_ID = 100
- SEND_WITH_CA_IND = N
- SEND_OFFER_OC_IND = Y

Entry 2:

- CUSTOMER_ACT_SEQ = 36
- OFFER_ACTIVITY_ID = 110
- SEND_WITH_CA_IND = Y
- SEND_OFFER_OC_IND = Y

Entry 3:

- CUSTOMER_ACT_SEQ = 36
- OFFER_ACTIVITY_ID = 120
- SEND_WITH_CA_IND = Y
- SEND_OFFER_OC_IND = N

Changes in Product Catalog to Edit Existing Versions

The *CM activity* general reference table (GRT) in version 5.5 has been replaced by a new GRT, *Customer activity*. The new GRT defines the relationship between activity codes and charge codes.

Because it is not possible to change item parameters in PITs when editing a Product Catalog version, the *CM activity* table must be changed.

To change the CM activity table:

In the ref area of the database:

1. In the PC1_CM_ACTIVITY table, edit the Activity code field.
2. Change the field's data from alphabetic to numeric.

note *The field's data type remains Varchar.*

For example:

Change CHANGEPP to 12.

The numeric value is sent to Billing.

Changes in Product Catalog to Create New Versions

The *CM activity* general reference table (GRT) in version 5.5 has been replaced by a new GRT, *Customer activity*. The new GRT defines the relationship between activity codes and charge codes. As a result, the *OC CM activity* PIT must be changed.

This includes the following activities:

- Adjusting the *OC CM Activity* PIT's Item Parameter
- Adjusting the Private Qualification criterion
- Adjusting the Change Calculate charge handler

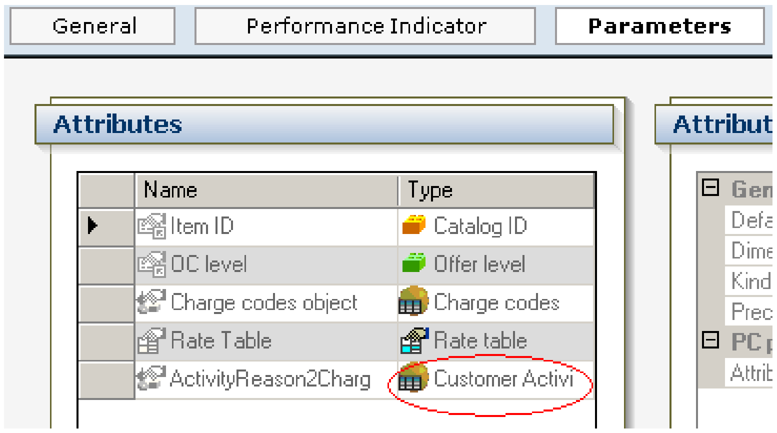
These activities are described in the following subsections.

Adjusting the OC CM Activity PIT's Item Parameter

To adjust the PIT's Item Parameter:

From the Product Catalog:

1. Lock the *OC CM activity* PIT.
2. On the Item Parameters tab, change the type of the ActivityReason2ChargeCode attribute from the *CM activity* GRT to the *Customer activities* GRT.



Name	Type
Item ID	Catalog ID
OC level	Offer level
Charge codes object	Charge codes
Rate Table	Rate table
ActivityReason2Charg	Customer Activi

Adjusting the Private Qualification Criterion

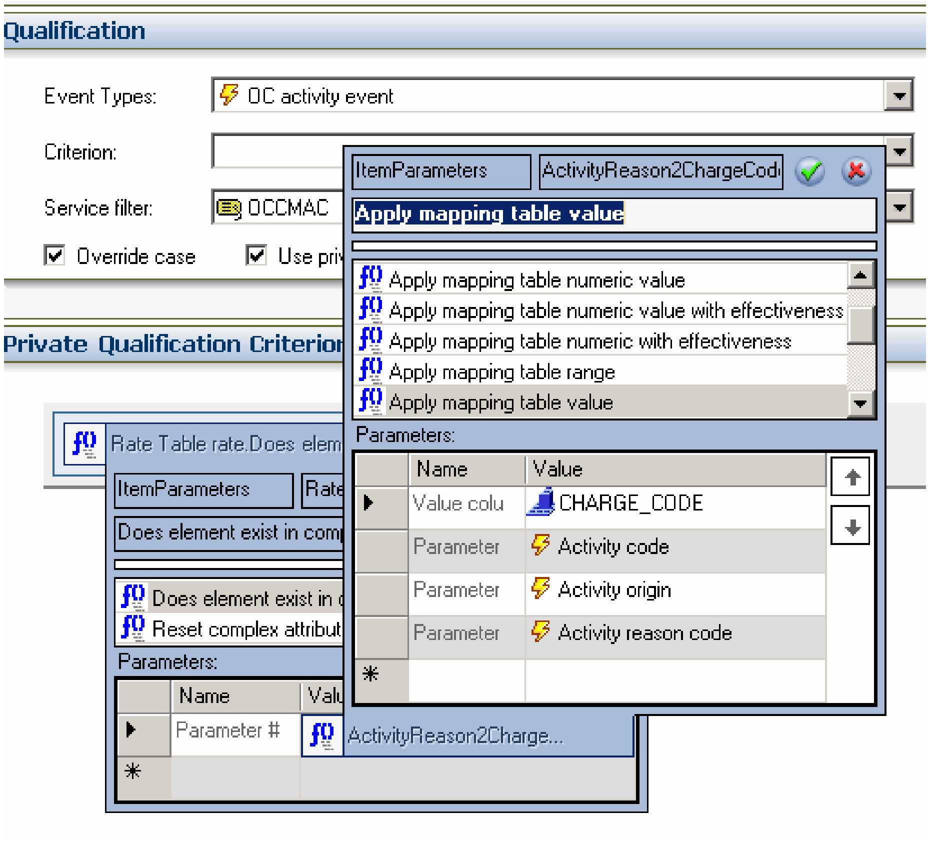
The private qualification criterion of *Event to be rated* must be changed in the Computation handler. The *Activity Origin* dynamic parameter must be added between the *Activity code* and *Activity reason code* parameters. The order of the parameters is important, since it reflects the order of the fields in the table's primary key.

To add the Activity Origin parameter:

From the Product Catalog:

1. Create a Horizontal correction.
2. In the Technical area, access the **Implementation Repository**.
3. Expand the Implementation Repository tree to the Pricing item types, to the **OC CM activity PIT**.
4. Open the Computation handler and select **Event to be handled**.

5. Under the General tab, select **Private Qualification Criterion**.
6. Double-click the **ActivityReason2ChargeCode** function.
7. Insert the **Activity origin** parameter between *Activity code* and *Activity reason code*.


Qualification
Event Types:
OC activity event
Criterion:
Service filter:
OCCMAC
Override case
Use pri
Private Qualification Criterion
Rate Table rate.Does elem
ItemParameters
Rate
Does element exist in com
Does element exist in
Reset complex attribut
Parameters:
Name
Valu
Parameter #
ActivityReason2Charge...
ItemParameters
ActivityReason2ChargeCod
Apply mapping table value
Apply mapping table numeric value
Apply mapping table numeric value with effectiveness
Apply mapping table numeric with effectiveness
Apply mapping table range
Apply mapping table value
Parameters:

Name	Value
Value colu	CHARGE_CODE
Parameter	Activity code
Parameter	Activity origin
Parameter	Activity reason code

Adjusting the Change Calculate Charge Handler

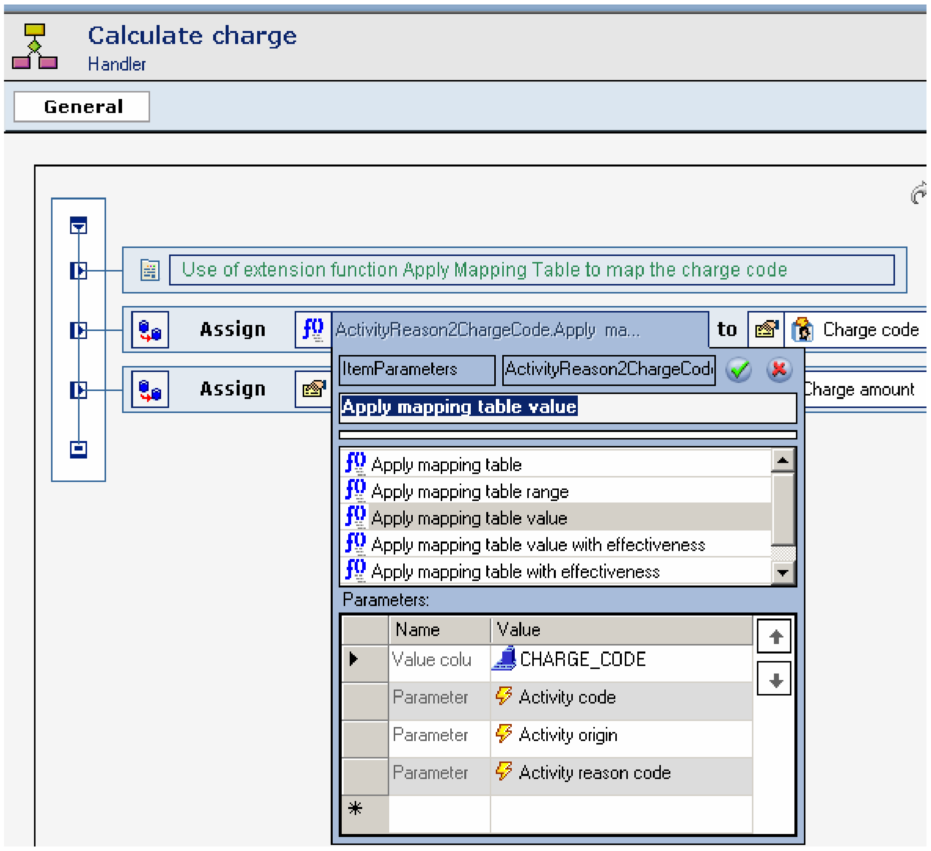
The *Activity Origin* dynamic parameter must be added between the *Activity code* and *Activity reason code* parameters. The order of the parameters is important, since it reflects the order of the fields in the table’s primary key.

To add the Activity Origin parameter:

From the Product Catalog:

1. Perform the same steps as outlined in “Adjusting the Private Qualification Criterion” in this chapter.
2. Under *Event to be handled*, select **Calculate charge**.
3. Double-click the **ActivityReason2ChargeCode** function.

4. Insert the **Activity origin** parameter between *Activity code* and *Activity reason code*.


Calculate charge
Handler
General
Use of extension function Apply Mapping Table to map the charge code
Assign
ActivityReason2ChargeCode.Apply ma...
to
Charge code
Assign
Charge amount
ItemParameters
ActivityReason2ChargeCode
Apply mapping table value
Apply mapping table
Apply mapping table range
Apply mapping table value
Apply mapping table value with effectiveness
Apply mapping table with effectiveness
Parameters:

Name	Value
Value colu	CHARGE_CODE
Parameter	Activity code
Parameter	Activity origin
Parameter	Activity reason code

Extract Tool Implementation

The following subsections describe the implementation upgrade of the Extract Tool. You must perform these procedures only if you need to maintain your version 5.5 customizations in version 6.0. These procedures also enable you to change your version 5.5 core references to point to version 6.0 as relevant.

note *Plan on allotting 15 to 30 minutes for these activities.*

Importing a New Core Repository

After the database has been converted to comply with version 6.0 (refer to "Running the Extract Tool Converter" in Chapter 6), it may be necessary for the customization component import the core implementation data that is valid for version 6.0.

If so, the customization component must use the Extract Tool to import the core implementation, which is delivered as an Extract Tool export file.

The export file is called:

ABP_V600_implementation_repository.etf

It is located in:

Extract Tool $install_dir/db/repository

This file is usually generated after the core implementation has been finalized. After the file has been created, it is distributed to the customization components. The components can import the contents of the export file into their repository, and update their implementations using old core objects in the new version.

The import process does not change any customization data; it only imports the new core data into the customization repository.

Updating Customization References

This step must only be performed by components that have based their report implementation on core objects, such as data objects, and want to use updated core objects. This is the final step in importing an export file into the repository.

note *Although this procedure updates references automatically, you can update references manually as necessary.*

To update customization references:

From the Extract Tool (just after import):

The Extract Tool displays a list of customization objects that reference core objects in an older version than the imported one.

- Select any references of the customization objects you whish to update with the new version of the core object.

After the correct references are selected, the Extract Tool updates the references from the old objects to the new ones, according to the following guidelines:

- If the referencing customization object is in the DRAFT status, the Extract Tool updates the object.
- If the referencing customization object is in the FINAL status, the Extract Tool first opens a new object version and then updates the new version to reference new objects.

8. POST-UPGRADE ACTIVITIES

Following the completion of the upgrade process, some post-upgrade activities are performed, such as releasing pending versions from the Product Catalog, starting up the applications, running full Update Handler extracts, performing sanity checks and making another back up of the database.

This section provides detailed step-by-step instructions to guide you through the activities to perform following the upgrade of Amdocs Billing platform from version 5.5 to version 6.0.

The following diagram illustrates these post-upgrade activities. Subsequent sections provide more details about each of these stages.

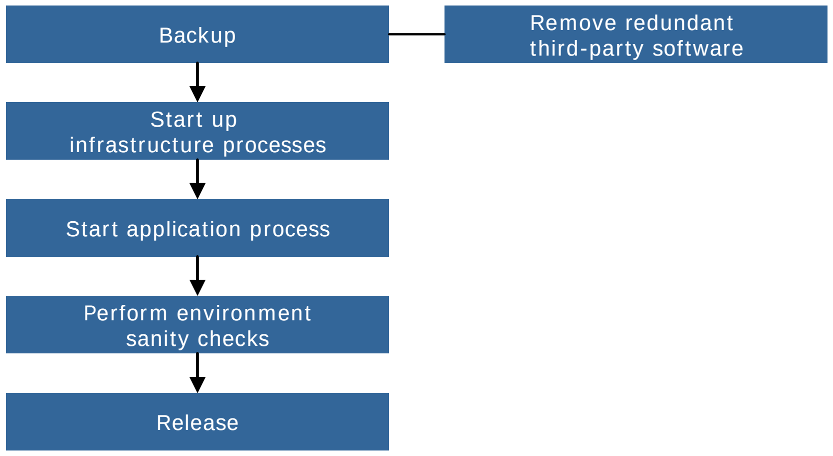

Backup
Remove redundant third-party software
Start up infrastructure processes
Start application process
Perform environment sanity checks
Release

Figure 8-1: Post-Upgrade Activities Workflow

Updating the Daemon Manager Configuration

In version 6.0, OpDaemon has been replaced by the Daemon Manager.

The Daemon Manager's configuration file is located in the AMC configuration directory (${MON_CONFIG_DIR}). Its naming convention is amc1_Daemon_${USER}.xml.

The configuration XML file is generated based on the entries in the GN1_TASK_CONNECT and GN1_CONNECT_PARAMS tables and the daemon templates.

The daemon templates and the SQL files containing the entries for the above GN tables may be located under any of the following:

- ABP_BIN
- ABP_CUSTOM_BIN
- ABP_APP_BIN
- ABP_APP_CUSTOM_BIN
- ABP_PRIVATE_ETC

All the daemon templates and SQL files are gathered into a single directory, in the order in which they are listed above, such that those that appear later in the list overwrite those that appear earlier.

The naming convention for the daemon templates is:

<module><layer>_Daemon_TMPL.xml

The GN1_TASK_CONNECT table contains the entries for all the instances of all the daemons. The GN1_CONNECT_PARAMS table provides the data required for running the specific instance (user, host, runtime parameters, and so on).

Use both these tables to make changes to the daemons' configuration (either manually or with the SQL files described above).

After changing the daemons' configuration, run the ATL1_ConfDmnMngr_Sh script. This generates a new configuration file for the Daemon Manager.

To generate the daemon XML file:

From the command line:

- In the work account, do one of the following:

  a. Manually edit the GN1_TASK_CONNECT and GN1_CONNECT_PARAMS tables.

  b. Run the ATL1_ConfDmnMngr_Sh script.

  -Or-

  c. Update the SQL files that populate the database with the relevant data. The naming convention for these SQL files is:

     <module><layer> _CoreDmnData.sql

  d. Run the following script:

     ATL1_OpGenUpdate_sh –storage

Removing Redundant Daemons and Jobs

A number of daemons and jobs have been removed in version 6.0.

To remove daemons and jobs:

From the command line:

1. In the database, remove the following daemons:
   - CM1UHUNLD
   - CM1UNLOAD
   - MF1ADMAIN
   - ORC1ALLPROC
   - ORC1PROC
   - RPR1CYCLE

2. Remove the following jobs:
   - AC1ADAPTERREP
   - AC1AUDMOTOR
   - AC1CLEAUPMAIN
   - AC1ENDOFDAY
   - AC1FLBALREP
   - AC1MOVTOCLEAN
   - AC1PROGRAMREP
   - AC1STAUMTOR
   - AC1TOKENREP
   - AR1GLEXT
   - BL1PCORCBRDG
   - BL1RATRACK
   - MF1FILETOQUE
   - MF1FTDOUTCOL
   - MF1MAINDRIVER
   - MF1RNMAINDR
   - OP1STARDMN
   - OP1STAREOD
   - OP1STARTJRL
   - OP1TUXDOWN
   - OP1TUXUP
   - PM1ALLRATER
   - PM1BILLEVEXT
   - PM1BILLPIEXT
   - PM1CHECKPOINT
   - PM1CMEXTRACT
   - PM1CYCLEOPEN
   - PM1EVENTEXT
   - PM1EVENTREC
   - PM1MANUALMARK
   - PM1NEWCYCLE
   - PM1NEWSTATE
   - PM1NURATER
   - PM1PIEXTRACT
   - PM1PIEXTREC
   - PM1PIMAINREC
   - PM1PREXTRACT

- PM1RNALLRATER
- PM1SPCHECKPT
- PM1STOPCMEXT
- PM1STOPRERATE
- PM1STOPREREXT
- PM1USQSTOP
- RM1ENDPSTATUS
- RM1ENDSTATUS
- RM1ENDSTSRP
- RM1EXTVALUES
- RM1RELEASELCK
- RPL1REFRESH
- RPL1TRBSUB
- VM1REFRESH

Performing Environment Sanity Checks

This activity is performed manually. Before it is performed, all other upgrade activities must be successfully completed.

- Perform sanity checks on the environment as described in *Amdocs Billing Platform Server Installation Guide (Runtime)*.

Starting Operational Daemons

The following subsections provide instructions for starting the Operational daemons:

- Daemon Manager
- CheckImmediate
- OpScheduler
- MRO

Step	Component / Module / Utility	Process	Procedure	Post-Activity	Remarks
1.	AMC	Daemon Manager	*From AMC:* Process Management > Daemons > Daemon Manager Monitor > Click AMCDaemon > Click Start.		
2.	AMC	CheckImmediate Daemon	*From AMC:* Process Management > Daemons > All Daemons Monitor > Click OP1CHKIMMED_1 > Click Start.		
3.	AMC	OpScheduler Daemon	*From AMC:* Process Management > Daemons > All Daemons Monitor > Click OP1SCHEDULER_1 > Click Start.		
4.	AMC	Message Router (MRO)	*From AMC:* Process Management > Daemons > All Daemons Monitor > Click OP1MRO_1 > Click Start.		

Starting Applicative Processes

The following subsections provide instructions for starting applicative processes.

Step	Component / Module / Utility	Process	Procedure	Post-Activity	Remarks
1.	Product Catalog	Converter	Refer to “Running the Product Catalog Converter” in Chapter 6.	None	
2.	File Loader	Population of the File Loader table	*From Screen Composer:* Run UTL1FILELD.	None	
3.	Product Catalog	Analyzer	*From AMC:* Billing > Ongoing Operations > Monitor & Control > Click PC Rate Change Analyzer Initiator > Start.		The PC Analyzer process performs the following updates: ■ Retrieves the new Product Catalog version from the BL1_XML_DISTRIB table. ■ Publishes a transaction with MEMBER_ID=2009 to the TRB1_PUB_LOG table. ■ Updates the BL1_XML_DISTRIB _CNTRL table for versions handled. Following successful completion, the process status changes to CO.

Step	Component / Module / Utility	Process	Procedure	Post-Activity	Remarks
4.	RTS		*From AMC:* Process Management > Daemons > All Daemons Monitor > Click UTL1RTS_AMC_1 > Start.	■ Verify that the RTS daemon is running: ● *From UNIX*, enter the command: psu | grep -i UTL1Rts* ● Verify that the STATUS in the UTL1_RTS_CONTROL reference table is WORKING ■ Verify that all entries in the UTL1_RTS_CHANGES table have a status of COMPLETE.	**Best Practice:** It is recommended to run the RTS each time the UTL1_RTS_CHANGES table is populated. Major milestones: ■ After the database upgrade ■ After Implementation upgrade ■ After Product Catalog distribution
5.	Reference Tables	Distribution to the Production area	Refer to “Distributing Reference Tables to the Production Area” in this chapter.	**This activity applies to production environments only.**	
6.	XML Distribution Process		*From Screen Composer:* Run CM1XMLOAD.	■ Verify that the following Customer Management tables are populated with the new offer parameters: ● CSM_OFFER ● CSM_OFFER_PARAM ● CSM_OFFER_ITEM ● CM1_OFFER_RELATION ■ Verify that the RELEASE_STATUS for all records in the *_XML_DISTRIB tables is USED.	Following each new release of Product Catalog versions, the XML Distribution process updates the Customer Management reference tables in order to update any changes made to offers.

Step	Component / Module / Utility	Process	Procedure	Post-Activity	Remarks
7.	A&F (Guiding)	Update Handler Full Unload	*From Screen Composer:* Run CM1UHFUNLD	Check the results: ■ Verify that three files were created in $ABP_UH_ROOT/data. These files contain relevant data that was mapped from the Customer Management Oracle tables. ■ In the AC1_CONTROL table, verify that three new records were created with the status RD. ■ Open /var/m3g/projs/uh/log and check the log file CM1UHFUNLD.log	
8.	A&F (Guiding)	Update Handler Full Load	*From Screen Composer:* Run CM1UHFLOAD	Check the results: ■ Verify that the following application database tables (Oracle or TimesTen) are populated with the contents of these files: ● GD1_SUBSCRIBER_DEST ● GD1_SUBSCR_KEY ● GD1_CUSTOMER_DETAILS ■ Open /var/m3g/projs/uh/log and check the log file CM1UHFLOAD.log.	

Step	Component / Module / Utility	Process	Procedure	Post-Activity	Remarks
9.	Rater	Update Handler Full Extract job	*From Screen Composer:* Run PM1CMGENEXT	Check the results: ■ Check that at least two XML files have been created for each cycle in $ABP_PM_ROOT/interfaces/input. For example: ● subscriberParametersF_1_90_20020123164010.xml ● subscriberOffersF_1_90_20020123164010.xml ■ In the BILL_CYCLE field of the CUSTOMER table, check the cycle to which the customer belongs. ■ In the AC1_CONTROL table, verify that new records were created with the status RD. (Two records should be created per billing cycle.)	It is highly recommended to run a full Rater Update Handler process for improved performance.
10.	Rater	Update Handler Full Load job	*From Screen Composer:* ■ Double-click Full or Incremental Load from XML Files ■ Populate the relevant JOB REQ. ■ Run PM1LOAD	Check the results: ■ Check that the loaded data matches the subscriber data in the following database tables (TimesTen or Oracle): ● CUSTOMER_PARAMETERS ● CUSTOMER_OFFERS	
11.	Billing	Transaction Listener for SOR	*From AMC:* Billing > Ongoing Operations > Monitor & Control > Click BLBTLSOR > Click Start.	Verify that the process adds and updates records in the following tables: ■ BL1_BLNG_ARRANGEMENT ■ BL1_CUSTOMER ■ BL1_PAY_CHANNEL ■ BL1_PAYER_CUST_REL (in cases of distribution)	

Step	Component / Module / Utility	Process	Procedure	Post-Activity	Remarks
12.	Billing	Cycle Request Listener	*From AMC:* Billing > Cycle Operations > Monitor & Control > Cycle Flows Initiators > Click RQSLSNR > Click Start.	Verify that the Cycle Request Listener is running: *From UNIX:* Enter the command: psu | grep bl1	
13.	Billing	Split and Merge Daemon	*From AMC:* Billing > Cycle Operations > Monitor & Control > Cycle Flows Initiators > Click SPLTMRG > Click Start.	Verify that the Split & Merge daemon is running: *From UNIX:* Enter the command: psu | grep bl1	
14.	Transaction Broker	Transaction Broker	*From AMC:* Transaction Broker > TRB Manager > Monitor & Control > Click TRB1ENGINE_<instance> > Click Start.	Verify that the Transaction Broker is running: *From UNIX:* Enter the command: psu | grep -i TRB	
15.	Transaction Broker	APInvoker	*From AMC:* Transaction Broker > TRB Manager > Monitor & Control > Click TLS1APINV_AMC > Click Start.	Verify that the APInvoker is running: *From UNIX:* Enter the command: psu | grep -i TRB	
16.	A&F (Guiding)	Update Handler Incremental Unload in Guiding	*From AMC:* Event Processing > AMC Master > A&F > Monitor & Control > Click UHUNLD > Click Start.	Check the results: ■ In the AC1_CONTROL table, verify that three new records were created with the status RD. ■ In the TRB1_SUB_LOG table, verify that there are no entries for SUB_APPL_ID=3005 ■ In the TRB1_SUB_ERRS table, verify that there are no entries for SUB_APPL_ID=3005	

Step	Component / Module / Utility	Process	Procedure	Post-Activity	Remarks
17.	A&F (Guiding)	Update Handler Incremental Load in Guiding	*From AMC:* Event Processing > AMC Master > A&F > Monitor & Control > Click UHLOAD > Click Start.	Check the results: ■ The following database tables (Oracle or TimesTen) are populated with the contents of these files: • GD1_SUBSCRIBER_DEST • GD1_SUBSCR_KEY • GD1_CUSTOMER_DETAILS ■ In the AC1_CONTROL table, verify that three new records were created with the status CO.	
18.	Rater	Update Handler Incremental Extract	*From AMC:* Event Processing > AMC Master > Rater > Monitor & Control > Click CMExtract > Click Start.	Check the results: ■ Check that at least two XML files have been created for each cycle in $ABP_PM_ROOT/interfaces/input. For example: • subscriberParametersI_1_90_20020123164010.xml • subscriberOffersI_1_90_20020123164010.xml. The files are created per cycle, so if there are transactions for activities to subscribers in more than one cycle, then two files are created per cycle. ■ In the TRB1_SUB_LOG table, verify that there are no entries for SUB_APPL_ID=3004 ■ In the TRB1_SUB_ERRS table, verify that there are no entries for SUB_APPL_ID=3004	

Step	Component / Module / Utility	Process	Procedure	Post-Activity	Remarks
19.	Rater	Update Handler Incremental Load	*From AMC:* Event Processing > AMC Master > Rater > Monitor & Control > Click CM2RaterLoad > Click Start.	Check the results: ■ In the AC1_CONTROL table, verify that the records created in the extract process have a status of CO. ■ Check that the loaded data matches the subscriber data in the following Oracle or TimesTen tables: ● CUSTOMER_PARAMETERS ● CUSTOMER_OFFERS	Select the daemon required for the relevant Partition ID.
20.	Online Charging	Formatting & Routing Server	*From AMC:* Online Charging > OLC > Monitor & Control > Click FR > Click Start.	Verify that the Formatting & Routing server is running: *From UNIX:* Enter the command: psu | grep -i olc	Select the daemon for the server that suits your configuration.
21.	Online Charging	Charging Engine Server	*From AMC:* Online Charging > OLC > Monitor & Control > Click RB > Click Start.	Verify that the Charging Engine server is running: *From UNIX:* Enter the command: psu | grep -i olc	Select the daemon for the server that suits your configuration.
22.	Online Charging	File Registration Daemon	*From AMC:* Online Charging > OLC > Monitor & Control > Click FRD > Click Start.	Verify that the File Registration daemon is running: *From UNIX:* Enter the command: psu | grep -i olc	
23.	Online Charging	Formatting & Routing Offline daemon (CDR Processing Engine)	*From AMC:* Online Charging > OLC > Monitor & Control > Click CDR > Click Start.	Verify that the Formatting & Routing Offline daemon (CDR Processing Engine) is running: *From UNIX:* Enter the command: psu | grep -i olc	

Step	Component / Module / Utility	Process	Procedure	Post-Activity	Remarks
24.	Online Charging	Session Expiration daemon	*From AMC:* Online Charging > OLC > Monitor & Control > Click SED > Click Start.	Verify that the Session Expiration daemon is running: *From UNIX:* Enter the command: psu | grep -i olc	
25.	Online Charging	Event Auditing daemon	*From AMC:* Online Charging > OLC > Monitor & Control > Click EAD > Click Start.	Verify that the Event Auditing daemon is running: *From UNIX:* Enter the command: psu | grep -i olc	
26.	A&F	File Listener	*From AMC:* Event Processing > AMC Master > A&F > Monitor & Control > Click MF1LSN > Click Start.	Check the results: Verify that the File Listener is running: *From UNIX:* Enter the command: psu | grep MF1	
27.	A&F	Main Driver	*From AMC:* Event Processing > AMC Master > A&F > Monitor & Control > Click MF1MD > Click Start.	Check the results: Verify that the Main Driver is running: *From UNIX:* Enter the command: psu | grep MF1	Select the Main Driver for the relevant instance.
28.	A&F	Reguiding	*From AMC:* Event Processing > AMC Master > A&F > Monitor & Control > Click MF1RGD > Click Start.	Verify that the Reguide process is running: From UNIX, enter the command: psu | grep MF1	

Step	Component / Module / Utility	Process	Procedure	Post-Activity	Remarks
29.	A&F	Postpaid Rater	*From AMC:* Event Processing > AMC Master > Rater > Rating > Monitor & Control > Click Rater > Click Start.	Verify that the Postpaid Rater is running: *From UNIX:* Enter the command: psu | grep pm1	Select the Rater for the relevant Partition ID.
30.	A&F	Dispatcher	*From AMC:* Event Processing > AMC Master > Rater > Monitor & Control > Click Dispatcher > Click Start.	Verify that the Dispatcher is running: *From UNIX:* Enter the command: psu | grep pm1	Select the Dispatcher for the relevant instance.

Distributing Reference Tables to the Production Area

This activity applies to production environments only.

To distribute reference tables to the Production area:

From Screen Composer:

- Run the following jobs in order:
  - UTL1RELEASE – Distributes reference tables from the Work area to the Wait area.
  - UTL1LOADBOH – Loads the business organization hierarchy (BOH) from the Product Catalog XML into the GN1_BOH_RELATION table. This job also sets the updated path for GN1_BE_VALID_VALUES into the table.
  - UTL1EXPBOH – Duplicates rows with BE in their primary key according to their business organization hierarchy.
  - UTL1DISTRIB – Distributes reference tables from the Wait area to the Read area, using a full snapshot process. This is done only by the production-line testing environment.
  - SYNSWITCH – Changes the synonym so that the application points to the updated Read area.

Removing Redundant Third-party Software

This activity is performed manually. It is not dependent upon the completion of any other activity.

- Remove any version 5.5 third-party software that is no longer used.

9. Testing Upgrade Guidelines

The aim of the Testing upgrade is to define a repeatable upgrade process that can be performed by accounts, and to provide guidelines and recommendations for this process.

This part of the upgrade is performed by the Infrastructure and Testing teams that are responsible for upgrading existing testing environments to the new version.

This chapter provides guidelines that enable a smooth upgrade to the version testing environment. This enables the testing team to migrate a full environment to the new version, thus supplying the basic test data to the remaining environments. In addition, the scenarios described here test the upgrade process and verifies that it covers all the changes made by the applications.

The upgrade process activities described is this chapter must be incorporated into the upgrade calendar.

Calendar Design Guidelines

The testing calendar must include tests and scenarios that are defined per application, and that are carefully tested after the upgrade is performed. The following types of tests are recommended:

- Minimum acceptance tests (MAT) performed on both the old and the new versions, using basic MAT scenarios to ensure that the basic functionality works before and after the upgrade. Minimum manipulations are performed on the upgraded data.
- Upgrade calendar tests performed on both versions using predefined scenarios, covering all core components, focusing on:
  - Continuity of scenarios before and after the upgrade
  - Activities and data manipulation on upgraded data
  - Functionality for the new version performed on upgraded data
- Upgrade calendar tests performed on both versions using predefined scenarios, covering customizations developed the customization team.
- Final regression tests resulting from the latest service pack for version 5.5 and upgraded to the final released version, using predefined scenarios remaining from the MAT and upgrade calendar tests, as well as some late functionality.
- Non-functional tests (NFT) to ensure readiness of the upgrade in a production-like environment with a high volume of data.
- High-availability tests to ensure upgrade of the production environment without downtime.

Recommended Upgrade Calendar Tests

It is recommended that the upgrade calendar include functional tests performed across applications. Examples of the recommended scenarios are described below, by application:

Service Management, Product Catalog and Reference Table Synchronizer

To test the correct functioning of Service Management, the Product Catalog and the Reference Table Synchronizer, it is recommended to create one pending version in Service Management, and one immediate and two or more pending versions in the Product Catalog prior to the upgrade.

Distribute the immediate versions before the upgrade and verify that reference tables are synchronized.

After the upgrade, convert Product Catalog data from version 5.5 to version 6.0, create a new service pack and release the pending versions created in version 5.5 from the Product Catalog.

Customer Management

To test the correct functioning of Customer Management, it is recommended to perform the following scenarios prior to the upgrade:

- Create Customer Management hierarchies
- Suspend some subscribers
- Cancel some subscribers
- Perform future activities: activation, cancellation, restore subscriber, suspend, move subscriber, add additional offer, change the price plan for subscribers on and after the upgrade date
- Reserve a subscriber
- Pre-activate a subscriber
- Create a memo

Following the upgrade, perform the following activities:

- Create additional Customer Management hierarchies
- Add subscribers to existing customers
- Suspend subscribers
- Cancel upgraded subscribers
- Restore subscribers that were suspended prior to the upgrade
- Renew and resume subscribers that were canceled prior to the upgrade
- Change customer and subscriber information, such as the name, address, and so on
- Make sure that all future activities are performed on schedule after the upgrade
- Perform a backdated activity, such as a change of price plan that enforces reguiding (for cycles that are closed after the upgrade)

- Move a subscriber from one customer to a customer created after the upgrade
- Perform a search by name and ID on upgraded data
- Change resource MSISDN and SIM
- Create a memo and view the memo list
- Perform a sale event to verify that the pre-activated subscriber is activated

Rating

To test the correct functioning of Rating, it is recommended to perform the following activities:

- Run various kinds of existing event types, such as voice and data events, prior to the upgrade. The same events are executed after the upgrade in order to compare the results received.
- Prepare scenarios for rerating and reguiding.

Following the upgrade, it is recommended to test the following activities:

- PI maintenance – Verify that the archive, purge and reconstruct functions are correct for upgraded Performance Indicators.
- Check that the correct amount was carried over for rolling offers.
- Check that the amounts for non-rolling offers were less prior to the upgrade.
- Perform rerating and reguiding.
- Perform usage queries to verify Performance Indicators were not damaged during the upgrade.
- PI and usage extract for unbilled usage

Resource Management

To test the correct functioning of Resource Management, it is recommended to perform a sanity check after the upgrade. For example:

- Check aging for a subscriber canceled before the upgrade
- Check aging of a reserved subscriber’s resources
- Load resources (MSISDNs)
- Perform a query on the resource list

Billing and the Billing Configurator

To test the correct functioning of Billing and the Billing Configurator, it is recommended to prepare the testing environment with some customers that are pre-billing and some that are post-billing, and to perform activities to compare results and verify that definitions in the Billing Configurator are saved and active. For example, prior to the upgrade perform the following activities:

- Check usage queries and billing screens
- Execute one billing cycle

Following the upgrade, perform the following activities:

- Check usage queries and billing screens to compare the results with those before the upgrade
- Execute two billing cycles for the population that was upgraded in the middle of the billing cycle month
- Ensure that discount calculation is correct after the upgrade
- Check recurring charge proration and additional one-time charges after the upgrade

Online Charging and Replenishment Management

To test the correct functioning of Online Charging and Replenishment Management, it is recommended to test the following activities prior to the upgrade:

- Perform a reservation request
- Perform error handling for erred events
- Perform a Recharge activity

Following the upgrade, test the following activities:

- Arrival to limit and exceed limit after upgrade
- Threshold activities
- Continue with the reservation request from before the upgrade
- Expire a session opened before the upgrade
- Perform error handling from the previous version
- Perform a Recharge activity
- Cancel a Recharge activity from the previous version

Security

Following the upgrade, change the password for an existing user and create a new user.

Voucher Management

To test the correct functioning of Voucher Management, it is recommended to test the following activities prior to the upgrade:

- Add voucher
- Unsuccessful voucher activations

Following the upgrade, test the following activities:

- Lock voucher with an unsuccessful attempt
- Load voucher with a successful attempt

Accounts Receivable

To test the correct functioning of Accounts Receivable, it is recommended to test the following activities prior to the upgrade:

- Payment input
- Direct Debit extract and payment posting

Following the upgrade, test the following activities:

- Payment backout for payments made before the upgrade
- Found transfer from old customer to new and vice versa
- Direct Debit reject

Provisioning

To test the correct functioning of Provisioning, it is recommended to test the population of Provisioning tables before and after the upgrade.

Audit & Control

Following the upgrade, generate Audit & Control reports, perform cleanup on all the populations created before and after the upgrade.

The following elements are not included in the scope of the upgrade tests:

- Collection
- Business Organization Hierarchy
- JMS
- End-of-Day map

Environments

In order to test the upgrade to version 6.0 at the package level, the following are required:

- Six environments installed by AIM for version 5.5
  - Three environments on the HP platform for the UMAT, Core Components and Final Regression calendars
  - Two environments on the AIX platform for the UMAT and Final Regression calendars
  - One environment on the SUN platform for the UMAT calendar
- Storage for each environment
- A separate Security server for each environment
- An AMC Master for each environment

A backup of the version 5.5 environment (preferably after creation of the population) should be saved separately to rollback the environment to the previous version, if required.

In addition to the system test environments, a component test environment should be allocated to each development team.

The following table lists the in-house tools to be used in the scope of testing the upgrade to version 6.0, since they are part of the product:

Tools	Responsible for Upgrade	Comments
AMC	Infrastructure	Used to run Acquisition & Formatting, Rating, the Transaction Broker, Online Charging, and Billing
Screen Composer	Infrastructure	Used to maintain operational, application, and reference tables
DDMU	Infrastructure	Used to view PIs and rated events
Expert	Infrastructure	Used to correct rejected events
		Use to create and maintain the database

Migration of Test Data

It is highly recommended that each account migrate the system test data provided in the Delivery Contents on a minimum number of environments and reinstall or rebuild the remaining environments. This ensures that minimal effort is spent on data migration, and a large number of system test environments can be prepared quickly for system test purposes.

The migration of test data refers to application, reference and operational data.

Appendix A. Known Validation Errors in the Product Catalog

This appendix describes several known validation errors in the Product Catalog.

Attribute Values Must be Valid Values

In Product Catalog 5.5, it was possible to enter a default value manually. In version 6.0, a new validation has been added: if an attribute is based on an elementary type with valid values, the default value of the attribute must be one of the valid values. In this case, a new value must be selected.

Value Provider for Property Types

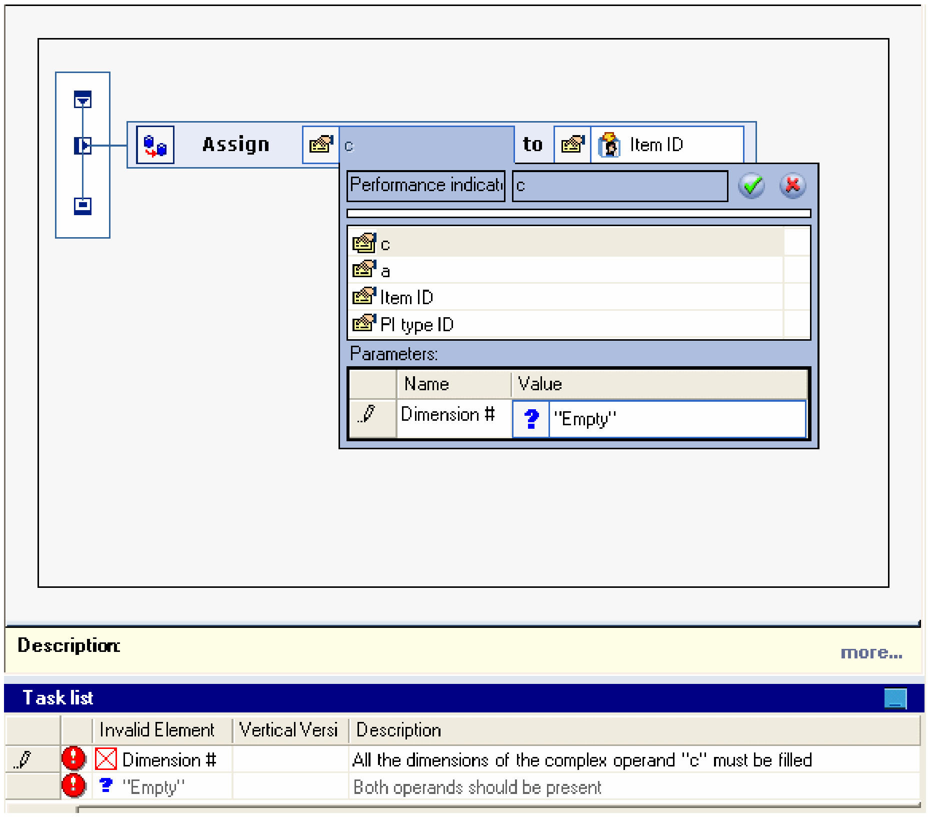
- The *Should be displayed* property type is defined with a value provider based on the *Yes/No indicator* elementary type. This means that all values for properties based on this property types must be set to Y, N or O (read-only).
- The *Parameter type ID* property type is defined with a value provider based on the *Parameter type* elementary type. This means that all property values must conform to the elementary type's valid values.

Deleting Empty Dimensions

In the previous version, empty dimensions in a statement were valid. In version 6.0, they are no longer valid and must be deleted. An example of such a validation:

To solve this validation, you must delete the empty dimension (new dimensions can be added when needed).

An example of such a validation:



Invalid Element	Vertical Versi	Description
Dimension #		All the dimensions of the complex operand "c" must be filled
"Empty"		Both operands should be present

Appendix B. Focal Point Table

This appendix provides blank tables to assist you in keeping track of the relevant focal points.

Internal Focal Points

Component/Area	Name	Phone/Cellular	Remarks
A&F			
ADBA			
AMC			
AR			
ART			
BC			
BL			
CC			
CM			
COL			
ET			
FE			
Generic Packages			
Infrastructure			
OLC			
OP			
PC			
PE			
Rater			
RIT/DDMU			
RM			
RPL			
SA (SEC)			
Screen Composer			
SM			
System Test			

Tiger			
TimesTen			
TRB			
UAMS Server			
UTL			
VM			

Other Focal Points

Component/Area	Name	Phone/Cellular	Remarks