



Bridging Transformation Enterprise - Wide

DISCOVERY REPORT

CITIE

Cambridge Technology Enterprises

A COST-EFFECTIVE METHOD FOR OPM UPGRADE VIA REIMPLEMENTATION

SUMMARY OF UPGRADE VS. RE-IMPLEMENT DECISION

The initial position of both Oracle and most GEMMS customers who wished to move into OPM on Oracle 11i was that a series of upgrade scripts presented the optimal path. The base release of Oracle 11i is sufficiently different from prior GEMMS and OPM11 versions that a series of upgrade scripts and processes were developed and offered by Oracle to the market at large. The biggest disadvantage was/is that they are very complex, and required a significant blackout period during the upgrade. First experiences with upgrading GEMMS to OPM 11i showed a blackout period of 70-90 hours.

Oracle attacked this blackout problem by first making it possible for (HP) users to upgrade to an 8.1.6 database prior to the upgrade of the applications. Subsequently, the complexity of the upgrade scripting and procedures themselves has been reduced. For many users, however, the blackout period is still a problem. There is data validation, scrubbing and reconfiguration which occurs in the middle of the process and which can affect the blackout schedule from customer to customer. The older the established instance of GEMMS, the larger this piece of the activity is likely to be.

Almost every GEMMS client looking at OPM 11i has considered re-implementation as an alternative. It permits the "raising" of a parallel 11i instance that can be patched and populated prior to cutover. It also allows purging of errors and revision of one-time-loaded configuration data from the old GEMMS instance before going forward on OPM 11i. After going live, the user can extract historic data from the old GEMMS instance at leisure and retain it in a data warehouse, then "kill" the old GEMMS instance.

A COST-EFFECTIVE METHOD FOR OPM UPGRADE VIA REIMPLEMENTATION

In early versions of OPM 11i, the API's available were the classic inventory and production tools from GEMMS. As a consequence, the migration of data en masse from a GEMMS instance into OPM was feasible (and sanctioned by Oracle) only with the use of a solution such as Crystallize or Fusion Technologies' toolsets. Typical cost for these solutions ran from \$200 - 300,000.

In October of 2001 Oracle delivered Family Pack 'G' with a new module encompassing Formula and Lab. Validity Rules were moved from the Formula and a new entity called a Recipe was introduced. At this time, a new Formula API and four API's to load Recipes were introduced to the product. In March, 2002 a new Production (Process Execution) Module with a new API was added to the product. In 2002 and 2003, a new and substantially more robust OPM Quality Module was released. API's to carry quality data from old to new are limited in their success.

Using the new API's and a tool such as DataLoader or WinRunner, it is now possible for most GEMMS users to do re-implementation using internal resources with minimal or no outside technical support. The cost of a re-implementation using these methods falls considerably below the cost as it was when 11i was first released. Following is what we have discovered while working with client M_____, who are taking this approach.

WHAT DATA, WHICH TOOL?

A list of the data to be converted from a legacy GEMMS into OPM follows, along with the method(s) we have used in the current engagement at M_____. The client is using two internal Business Analysts to extract data from GEMMS 4.10.08 (one Division) and BPICS (a second Division) into Excel spreadsheets. The spreadsheets are then "scrubbed" by users to eliminate bad data, add new required and desired data, etc. Those spreadsheets which are going to be inserted using API's or SQL scripts are then converted to a series of flat files, and the rest are formatted for DataLoader (client's tool of choice).

- ➔ **Item Master Records** - 2 API's plus 1 SQL script - For this exercise, spreadsheet work also included an Item ID conversion, as a new numbering system was adopted. The first API loads base item data; the second is needed to insert numeric conversion factor for dual UOM items. As soon as the base data was loaded, the names of IC_TRAN_PND and IC_TRAN_CMP were changed to prevent any transaction activity from "burning" the new record before conversion data could be loaded. The GL Business Class will not load via API, and an SQL script was written to insert it into the table.
- ➔ **Discrete Inventory Data** - The client was converting from an in-house Order Management system to Oracle OM, with telesales, iStore, complex sales promotions, etc. We created many item attribute flexfields in the Discrete Inventory Module to accommodate these needs. They were loaded into the new instance using the Discrete API.
- ➔ **Routing Activities and Resources** - M_____ decided to do a manual load of Activities and Resources, since there were not a large number of values. A group of four users populated the data in about two hours.
- ➔ **Routing Operations** - DataLoader was used to input this data via automated form entry.
- ➔ **Routings** - DataLoader input; in M_____ instance, there are four production facilities using Operations, Routings, Formulas, and Recipes.

- ➔ **Formulas** - Loaded using the Formula API. It took some work to get this right.
- ➔ **Recipes** - The original intent was to use the API's for this load. There are four of them, and the data is fairly complex. We discovered enough while attempting this to actually load some of the recipes accurately, but ran out of time. The result was that we loaded Recipe Headers via API, then manually populated them with the validity data (formula and version, routing and version, organization, and recipe use - production and costing). While Routing Class is not a required field on the Routings, the Recipe API treats it as such - with a special data restriction.
- ➔ **MAC Mapping** - Mapping was developed on spreadsheets and input using DataLoaders.
- ➔ **Item Costs** - Raw Material costs were applied using a DataLoader. After costs were input and all Recipe information had been applied, a Cost Rollup and Cost Update were performed in the new instance.
- ➔ **Inventory Balances** - We intend to use one or more of the inventory API's for this load. Almost all items are lot/location controlled with grade, status, and expiry date. We are currently analyzing just how we will accomplish this load in the short time allowed during the system cutover.

We have used two client and one contract technical resources to write extraction scripts and wrappers, in addition to the Business Analysts previously mentioned. Because of the complexity of the work required in Oracle Discrete Inventory required by the interfaces with OM and Purchasing, an outside consulting resource was used to develop Categories, Category Sets, and Item Attribute flexfields.

TIMING AND DATA INTEGRITY

The time to perform an actual conversion is about 7-8 days for all of the static or semi-static data (everything except inventory balances). This assumes that the new instance is already populated with all setup data. The client has about 5,500 Item Master records, and about 2,200 Formulas and Routings. Product Structure is often 6-8 layers "deep."

All converted data was first loaded to a "conversion" (CNV) instance; in the case of API's it was loaded with a "no commit" initially, then errors were identified and "scrubbed." Following this the data went into the CNV instance complete. Limited functional unit testing was performed in CNV, followed by a load into a CRP instance. Process testing in CRP included MRP, Production Batch Execution, and Cost Rollup. Finally, the Production instance was created.

SUMMARY

OPM re-implementation using the method outlined above is possible, controllable, and less expensive than the prior options. It does have some problems and "quirks" associated with it, and requires considerable forethought. The new Recipe API's are largely untried, and we did encounter some difficulties with them. Multiple iterations of API's and script are needed to completely load an Item Master record. The loading process needs to be well orchestrated in order to avoid errors and excessive conversion time.

We found any number of "dirty" data elements in the prior GEMMS instance, the product of three plus years of daily operation by many people. The "scrubbing" process with converted data was a good opportunity to both clean up the instance and identify education/training needs in the client organization.

All in all, this approach to an OPM 11i migration has a lot to recommend it, and not too many disadvantages.