Oracle® Process Manufacturing

Process Execution API User’s Guide

Release 11i

Part No. A97388-02

January 2003

Oracle Process Manufacturing Process Execution API User's Guide, Release 11i

Part No. A97388-02

Primary Author: Michele-Andrea Fields

Contributors: Bill Stearns, Thomas Daniel, Shrikant Nene

Contents

Send Us Your Comments

Oracle Process Manufacturing Process Execution API User's Guide, Release 11*i*

Part No. A97388-02

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

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

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

- FAX: 650-506-7200 Attn: Oracle Process Manufacturing
- Postal service:
  Oracle Corporation
  Oracle Process Manufacturing
  500 Oracle Parkway
  Redwood City, CA 94065
  U.S.A.
- Electronic mail message to appsdoc_us@oracle.com

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

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

Preface

Audience for This Guide

Welcome to Release 11*i* of the *Oracle Process Manufacturing Process Execution API User's Guide*.

This guide assumes you have a working knowledge of the following:

- The principles and customary practices of your business area.
- Oracle Process Manufacturing Process Execution APIs
- The Oracle Applications graphical user interface.

  To learn more about the Oracle Applications graphical user interface, read the *Oracle Applications User's Guide*.

See Other Information Sources for more information about Oracle Applications product information.

How To Use This Guide

This guide contains the information you need to understand and use Oracle Process Manufacturing Process Execution APIs.

- Chapter 1 describes how APIs are used, the basic business need of APIs, and the different Process Execution APIs offered.
- Chapter 2 describes how to use the Process Execution APIs.
- Chapter 3 describes the technical aspect of the APIs.
- Chapter 4 describes the business objects for each API.
- Appendix A describes messages and error codes.

- Appendix B lists the full GMEPAPIS.pls file.
- A Glossary provides definitions of terms that are used in this guide.

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible, with good usability, to the disabled community. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Standards will continue to evolve over time, and Oracle Corporation is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For additional information, visit the Oracle Accessibility Program Web site at http://www.oracle.com/accessibility/.

Accessibility of Code Examples in Documentation

JAWS, a Windows screen reader, may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, JAWS may not always read a line of text that consists solely of a bracket or brace.

Other Information Sources

You can choose from many sources of information, including online documentation, training, and support services, to increase your knowledge and understanding of Oracle Process Manufacturing System Administration.

If this guide refers you to other Oracle Applications documentation, use only the Release 11*i* versions of those guides.

Online Documentation

All Oracle Applications documentation is available online (HTML or PDF).

- **Online Help** - The new features section in the HTML help describes new features in 11i. This information is updated for each new release of Oracle Process Manufacturing System Administration. The new features section also includes information about any features that were not yet available when this guide was printed. For example, if your administrator has installed software from a mini-packs an upgrade, this document describes the new features. Online help patches are available on MetaLink.

- **11i Features Matrix** - This document lists new features available by patch and identifies any associated new documentation. The new features matrix document is available on MetaLink.
- **Readme File** - Refer to the readme file for patches that you have installed to learn about new documentation or documentation patches that you can download.

Related User's Guides

Oracle Process Manufacturing Process Execution APIs share business and setup information with other Oracle Applications products. Therefore, you may want to refer to other user's guides when you set up and use Oracle Process Manufacturing Process Execution APIs.

You can read the guides online by choosing Library from the expandable menu on your HTML help window, by reading from the Oracle Applications Document Library CD included in your media pack, or by using a Web browser with a URL that your system administrator provides.

If you require printed guides, you can purchase them from the Oracle Store at http://oraclestore.oracle.com.

Guides Related to All Products

Oracle Applications User's Guide

This guide explains how to enter data, query, run reports, and navigate using the graphical user interface (GUI) available with this release of Oracle Process Manufacturing System Administration (and any other Oracle Applications products). This guide also includes information on setting user profiles, as well as running and reviewing reports and concurrent processes.

You can access this user's guide online by choosing "Getting Started with Oracle Applications" from any Oracle Applications help file.

User Guides Related to This Product

Accounting Setup User's Guide

The OPM Accounting Setup application is where users set up global accounting attributes about the way financial data will be collected by OPM. These attributes include such things as account keys, financial calendars, and account segments.

Since OPM is closely integrated with Oracle General Ledger (GL), much of the attributes are defined in the Oracle GL instead of OPM, and therefore, the windows are display only within OPM. The *Oracle Process Manufacturing Accounting Setup User's Guide* describes how to setup and use this application.

Cost Management User's Guide

The OPM Cost Management application is used by cost accountants to capture and review the manufacturing costs incurred in their process manufacturing businesses. The *Oracle Process Manufacturing Cost Management User's Guide* describes how to setup and use this application.

Manufacturing Accounting Controller User's Guide

The Manufacturing Accounting Controller application is where users define the impact of manufacturing events on financials. For example, event RCPT (Inventory Receipts) results in a debit to inventory, a credit to accrued accounts payable, a debit or a credit to purchase price variance, etc. These impacts are predefined in the Manufacturing Accounting Controller application so users may begin using OPM to collect financial data out-of-the-box, however, they may also be adjusted per your business needs. The *Oracle Process Manufacturing Manufacturing Accounting Controller User's Guide* describes how to setup and use this application.

Oracle Financials Integration User's Guide

Since OPM is closely integrated with Oracle General Ledger, financial data that is collected about the manufacturing processes must be transferred to the Oracle Financials applications. The OPM Oracle Financials Integration application is where users define how that data is transferred. For example, users define whether data is transferred real time or batched and transferred at intervals. The *Oracle Process Manufacturing Oracle Financials Integration User's Guide* describes how to setup and use this application.

Inventory Management User's Guide

The OPM Inventory Management application is where data about the items purchased for, consumed during, and created as a result of the manufacturing process are tracked. The *Oracle Process Manufacturing Inventory Management User's Guide* includes information to help you effectively work with the Oracle Process Manufacturing Inventory application.

Physical Inventory User's Guide

Performing physical inventory count is the most accurate way to get an accounting of all material quantities purchased, manufactured, and sold, and update your onhand quantities accordingly. The OPM Physical Inventory application automates and enables the physical inventory process. The *Oracle Process Manufacturing Physical Inventory User's Guide* describes how to setup and use this application.

Order Fulfillment User's Guide

The OPM Order Fulfillment application automates sales order entry to reduce order cycle time. Order Fulfillment enables order entry personnel to inform customers of scheduled delivery dates and pricing. The *Oracle Process Manufacturing Order Fulfillment User's Guide* describes how to setup and use this application.

Purchase Management User's Guide

OPM Purchase Management and Oracle Purchasing combine to provide an integrated solution for Process Manufacturing. Purchase orders are entered in Oracle Purchasing and received in OPM. Then, the receipts entered in OPM are sent to Oracle Purchasing. The *Oracle Process Manufacturing Purchase Management User's Guide* describes how to setup and use this integrated solution.

Using Oracle Order Management with Process Inventory Guide

Oracle Process Manufacturing and Oracle Order Management combine to provide an integrated solution for process manufacturers. The manufacturing process is tracked and handled within Oracle Process Manufacturing, while sales orders are taken and tracked in Oracle Order Management. Process attributes, such as dual UOM and lot control, are enabled depending on the inventory organization for the item on the sales order. Order Management accepts orders entered through Oracle Customer Relationship Management (CRM). Within CRM, orders can originate from TeleSales, Sales Online, and iStore, and are booked in Order Management, making the CRM suite of products available to Process customers, through Order Management. The *Oracle Order Management User's Guide* and *Using Oracle Order Management with Process Inventory Guide* describes how to setup and use this integrated solution.

Process Execution User's Guide

The OPM Process Execution application lets you track firm planned orders and production batches from incoming materials through finished goods. Seamlessly integrated to the Product Development application, Process Execution lets you convert firm planned orders to single or multiple production batches, allocate ingredients, record actual ingredient usage, and then complete and close production batches. Production inquiries and preformatted reports help you optimize inventory costs while maintaining a high level of customer satisfaction with on-time delivery of high quality products. The *OPM Process Execution User's Guide* presents overviews of the tasks and responsibilities for the Production Supervisor and the Production Operator. It provides prerequisite setup in other applications, and details the windows, features, and functionality of the OPM Process Execution application.

Integration with Advanced Planning and Scheduling User's Guide

Oracle Process Manufacturing and Oracle Advanced Planning and Scheduling (APS) combine to provide an integrated solution for process manufacturers that can help increase planning efficiency. The integration provides for constraint-based planning, performance management, materials management by exception, mixed mode manufacturing that enables you to choose the best method to produce each of your products, and combine all of these methods within the same plant/company. The *Oracle Process Manufacturing Integration with Advanced Planning and Scheduling User's Guide* describes how to setup and use this application.

MPS/MRP and Forecasting User's Guide

The Oracle Process Manufacturing Material Requirements Planning (MRP) application provides long-term "views" of material demands and projected supply actions to satisfy those demands. The Master Production Scheduling (MPS) application lets you shorten that view to a much narrower and immediate time horizon, and see the immediate effects of demand and supply actions. The *Oracle Process Manufacturing MPS/MRP and Forecasting User's Guide* describes how to setup and use this application.

Capacity Planning User's Guide

The OPM Capacity Planning User's Guide describes the setup required to use OPM with the Oracle Applications Advanced Supply Chain Planning solutions. In addition, Resource setup, used by the OPM Production Execution and New Product Development applications, is also described.

Using Oracle Process Manufacturing with Oracle Manufacturing Scheduling

Oracle Process Manufacturing integrates with Oracle Manufacturing Scheduling to manage and utilize resources and materials. Through the Process Manufacturing application, you set up manufacturing, inventory, procurement and sales order data. Through the Manufacturing Scheduling application, you can optimize the schedule based on resource and component constraints and user predefined priorities. Using different optimization objectives, you can tailor Manufacturing Scheduling to meet your needs.

Using Oracle Manufacturing Scheduling helps you improve productivity and efficiency on your shop floor. By optimally scheduling shop floor jobs, and being able to quickly react to unplanned constraints, you can lower manufacturing costs, increase resource utilization and efficiency, and increase customer satisfaction through improved on-time delivery. The *Using Oracle Process Manufacturing with Oracle Manufacturing Scheduling User's Guide* describes how to setup and use this integrated solution.

Product Development User's Guide

The Oracle Process Manufacturing Product Development application provides features to manage formula and laboratory work within the process manufacturing operation. It lets you manage multiple laboratory organizations and support varying product lines throughout the organization. You can characterize and simulate the technical properties of ingredients and their effects on formulas. You can optimize formulations before beginning expensive laboratory test batches. Product Development coordinates each development function and enables a rapid, enterprise-wide implementation of new products in your plants. The *Oracle Process Manufacturing Product Development User's Guide* describes how to setup and use this application.

Quality Management User's Guide

The Oracle Process Manufacturing Quality Management application provides features to test material sampled from inventory, production, or receipts from external suppliers. The application lets you enter specifications and control their use throughout the enterprise. Customized workflows and electronic record keeping automate plans for sampling, testing, and result processing. You can compare specifications to assist in regrading items, and match customer specifications. Aggregate test results and print statistical assessments on quality certificates. Several preformatted reports and inquiries help manage quality testing and reporting. The *Oracle Process Manufacturing Quality Management User's Guide* describes how to set up and use this application.

Regulatory Management User's Guide

The Oracle Process Manufacturing Regulatory Management application generates the Material Safety Data Sheets (MSDSs) required by authorities to accompany hazardous materials during shipping. You can create MSDSs from OPM Formula

Management with Regulatory or Production effectivities. The *Oracle Process Manufacturing Regulatory Management User's Guide* describes how to setup and use this application.

Implementation Guide

The *Oracle Process Manufacturing Implementation Guide* offers information on setup. That is, those tasks you must complete following the initial installation of the Oracle Process Manufacturing software. Any tasks that must be completed in order to use the system out-of-the-box are included in this manual.

System Administration User's Guide

Much of the System Administration duties are performed at the Oracle Applications level, and are therefore described in the *Oracle Applications System Administrator's Guide.* The *Oracle Process Manufacturing System Administration User's Guide* provides information on the few tasks that are specific to OPM. It offers information on performing OPM file purge and archive, and maintaining such things as responsibilities, units of measure, and organizations.

API User's Guides

Public Application Programming Interfaces (APIs) are available for use with different areas of the Oracle Process Manufacturing application. APIs make it possible to pass information into and out of the application, bypassing the user interface. Use of these APIs is documented in individual manuals such as the *Oracle Process Manufacturing Inventory API User's Guide, Oracle Process Manufacturing Process Execution API User's Guide, Oracle Process Manufacturing Product Development Formula API User's Guide, Oracle Process Manufacturing Product Development Recipe API User's Guide, Oracle Process Manufacturing Quality Management API User's Guide*, and the *Oracle Process Manufacturing Cost Management API User's Guide.* Additional API User's Guides are periodically added as additional public APIs are made available.

Installation and System Administration

Oracle Applications Concepts

This guide provides an introduction to the concepts, features, technology stack, architecture, and terminology for Oracle Applications Release 11*i*. It provides a useful first book to read before an installation of Oracle Applications. This guide also introduces the concepts behind Applications-wide features such as Business Intelligence (BIS), languages and character sets, and Self-Service Web Applications.

Installing Oracle Applications

This guide provides instructions for managing the installation of Oracle Applications products. In Release 11*i*, much of the installation process is handled using Oracle Rapid Install, which minimizes the time to install Oracle Applications, the Oracle8 technology stack, and the Oracle8*i* Server technology stack by automating many of the required steps. This guide contains instructions for using Oracle Rapid Install and lists the tasks you need to perform to finish your installation. You should use this guide in conjunction with individual product user's guides and implementation guides.

Upgrading Oracle Applications

Refer to this guide if you are upgrading your Oracle Applications Release 10.7 or Release 11.0 products to Release 11*i*. This guide describes the upgrade process and lists database and product-specific upgrade tasks. You must be either at Release 10.7 (NCA, SmartClient, or character mode) or Release 11.0, to upgrade to Release 11*i*. You cannot upgrade to Release 11*i* directly from releases prior to 10.7.

Maintaining Oracle Applications

Use this guide to help you run the various AD utilities, such as AutoUpgrade, AutoPatch, AD Administration, AD Controller, AD Relink, License Manager, and others. It contains how-to steps, screenshots, and other information that you need to run the AD utilities. This guide also provides information on maintaining the Oracle applications file system and database.

Oracle Applications System Administrator's Guide

This guide provides planning and reference information for the Oracle Applications System Administrator. It contains information on how to define security, customize menus and online help, and manage concurrent processing.

Oracle Alert User's Guide

This guide explains how to define periodic and event alerts to monitor the status of your Oracle Applications data.

Oracle Applications Developer's Guide

This guide contains the coding standards followed by the Oracle Applications development staff. It describes the Oracle Application Object Library components needed to implement the Oracle Applications user interface described in the *Oracle Applications User Interface Standards for Forms-Based Products*. It also provides information to help you build your custom Oracle Forms Developer 6*i* forms so that they integrate with Oracle Applications.

Oracle Applications User Interface Standards for Forms-Based Products

This guide contains the user interface (UI) standards followed by the Oracle Applications development staff. It describes the UI for the Oracle Applications products and how to apply this UI to the design of an application built by using Oracle Forms.

Other Implementation Documentation

Oracle Applications Product Update Notes

Use this guide as a reference for upgrading an installation of Oracle Applications. It provides a history of the changes to individual Oracle Applications products between Release 11.0 and Release 11*i*. It includes new features, enhancements, and changes made to database objects, profile options, and seed data for this interval.

Multiple Reporting Currencies in Oracle Applications

If you use the Multiple Reporting Currencies feature to record transactions in more than one currency, use this manual before implementing Oracle Process Manufacturing System Administration. This manual details additional steps and setup considerations for implementing Oracle Process Manufacturing System Administration with this feature.

Multiple Organizations in Oracle Applications

This guide describes how to set up and use Oracle Process Manufacturing System Administration with Oracle Applications' Multiple Organization support feature, so you can define and support different organization structures when running a single installation of Oracle Process Manufacturing System Administration.

Oracle Workflow Guide

This guide explains how to define new workflow business processes as well as customize existing Oracle Applications-embedded workflow processes.You also use this guide to complete the setup steps necessary for any Oracle Applications product that includes workflow-enabled processes.

Oracle Applications Flexfields Guide

This guide provides flexfields planning, setup and reference information for the Oracle Process Manufacturing System Administration implementation team, as well as for users responsible for the ongoing maintenance of Oracle Applications product data. This manual also provides information on creating custom reports on flexfields data.

Oracle eTechnical Reference Manuals

Each eTechnical Reference Manual (eTRM) contains database diagrams and a detailed description of database tables, forms, reports, and programs for a specific Oracle Applications product. This information helps you convert data from your existing applications, integrate Oracle Applications data with non-Oracle applications, and write custom reports for Oracle Applications products. Oracle eTRM is available on Metalink

Oracle Manufacturing APIs and Open Interfaces Manual

This manual contains up-to-date information about integrating with other Oracle Manufacturing applications and with your other systems. This documentation includes API's and open interfaces found in Oracle Manufacturing.

Oracle Order Management Suite APIs and Open Interfaces Manual

This manual contains up-to-date information about integrating with other Oracle Manufacturing applications and with your other systems. This documentation includes API's and open interfaces found in Oracle Order Management Suite.

Oracle Applications Message Reference Manual

This manual describes all Oracle Applications messages. This manual is available in HTML format on the documentation CD-ROM for Release 11i.

Training and Support

Training

Oracle offers a complete set of training courses to help you and your staff master Oracle Process Manufacturing Process Execution APIs and reach full productivity quickly. These courses are organized into functional learning paths, so you take only those courses appropriate to your job or area of responsibility.

You have a choice of educational environments. You can attend courses offered by Oracle University at any one of our many Education Centers, you can arrange for our trainers to teach at your facility, or you can use Oracle Learning Network (OLN), Oracle University's online education utility. In addition, Oracle training professionals can tailor standard courses or develop custom courses to meet your needs. For example, you may want to use your organization structure, terminology, and data as examples in a customized training session delivered at your own facility.

Support

From on-site support to central support, our team of experienced professionals provides the help and information you need to keep Oracle Process Manufacturing Process Execution APIs working for you. This team includes your Technical Representative, Account Manager, and Oracle's large staff of consultants and support specialists with expertise in your business area, managing an Oracle8*i* server, and your hardware and software environment.

Do Not Use Database Tools to Modify Oracle Applications Data

Oracle STRONGLY RECOMMENDS that you never use SQL*Plus, Oracle Data Browser, database triggers, or any other tool to modify Oracle Applications data unless otherwise instructed.

Oracle provides powerful tools you can use to create, store, change, retrieve, and maintain information in an Oracle database. But if you use Oracle tools such as SQL*Plus to modify Oracle Applications data, you risk destroying the integrity of your data and you lose the ability to audit changes to your data.

Because Oracle Applications tables are interrelated, any change you make using Oracle Applications can update many tables at once. But when you modify Oracle Applications data using anything other than Oracle Applications, you may change a row in one table without making corresponding changes in related tables. If your tables get out of synchronization with each other, you risk retrieving erroneous information and you risk unpredictable results throughout Oracle Applications.

When you use Oracle Applications to modify your data, Oracle Applications automatically checks that your changes are valid. Oracle Applications also keeps track of who changes information. If you enter information into database tables using database tools, you may store invalid information. You also lose the ability to track who has changed your information because SQL*Plus and other database tools do not keep a record of changes.

About Oracle

Oracle Corporation develops and markets an integrated line of software products for database management, applications development, decision support, and office automation, as well as Oracle Applications, an integrated suite of more than 160 software modules for financial management, supply chain management, manufacturing, project systems, human resources and customer relationship management.

Oracle products are available for mainframes, minicomputers, personal computers, network computers and personal digital assistants, allowing organizations to integrate different computers, different operating systems, different networks, and even different database management systems, into a single, unified computing and information resource.

Oracle is the world's leading supplier of software for information management, and the world's second largest software company. Oracle offers its database, tools, and applications products, along with related consulting, education, and support services, in over 145 countries around the world.

Your Feedback

Thank you for using Oracle Process Manufacturing Process Execution APIs and this user's guide.

Oracle values your comments and feedback. At the end of this guide is a Reader's Comment Form you can use to explain what you like or dislike about Oracle Process Manufacturing Process Execution APIs or this user's guide. Mail your comments to the following address or call us directly at (650) 506-7000.

Oracle Applications Documentation Manager
Oracle Corporation
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Or, send electronic mail to **appsdoc_us@oracle.com**.

1 API Introduction

This document describes the Application Program Interfaces (APIs) that support external interfaces to Oracle Process Manufacturing (OPM) Process Execution application. The topics discussed in this chapter are:

- Introducing the Process Execution APIs
- Basic Business Needs
- Major Features
- Process Execution API Bill of Materials

Introducing the Process Execution APIs

Process Execution APIs let you import information from another system into the OPM Process Execution tables. When you import this information you can include all pertinent information using a tool that does not have cryptic IDs and system specific information. The interface ensures that your imported data contain the same detail as those you enter manually on the OPM Process Execution windows.

This document describes the usage of the business objects that are stored as PL/SQL packages within the OPM database schema, such as:

- Stored procedures used within these packages
- Parameters that these procedures accept and the values that return to the calling program
- Multilingual support
- Error handling methodology

What Is In This Document

This document describes the basic business needs, major features, architecture, and components for the Process Execution APIs insert, update, and delete features. The application is divided into application-specific objects that let you link OPM functionality into your own programs. The interfaces can make use of the standard functionality and logic implemented in the Process Execution application.

Process Execution APIs are written in PL/SQL that can be called by your own programs. To make use of these APIs, code your wrapper function that passes the appropriate parameters to the APIs. Your program is responsible for connecting to a database before calling an API function, and disconnecting from the database upon return. You can choose to write log files before calling and after returning from a function. If there is a problem during execution of a call, then the APIs return one of the following status codes:

- S for success
- E for error
- U for unknown or unexpected status
- N for item requiring a location
- V for inventory shortage exists
- I for incomplete manual transactions exist
- G for over allocation exists

Process Execution API Support Policy

Process Execution APIs are supported by Oracle. This means:

- Oracle provides objects and libraries needed to use the APIs and the documentation for their use.
- Oracle ensures that the APIs function as designed.
- Oracle does not support customer generated programs that use the APIs.

Technical Requirements

Process Execution APIs make use of the following standard Oracle Applications packages:

- FND_API - the standard Oracle Applications API version checking function. This is used by the stored procedure to check for a valid API version number and also contains constant variables such as TRUE and FALSE.
- FND_MESSAGE - the standard Oracle Applications messaging function. This is used by the stored procedure to report status and error handling.
- FND_PUB_MSG - the standard Oracle Applications message retrieval function used to search the procedure messages.

These packages are installed as part of the current release. Refer to the *Oracle Applications Coding Standards* manual for additional details.

Technical Overview

Process Execution APIs are designed to operate in an OPM 11i environment only. They offer the following basic API functionality:

- Creating, updating, and deleting information.
- Proper encapsulation.
- Synchronous processing following the business hierarchy.
- Detailed and translatable error messages.

Basic Business Needs

These APIs let you feed information from shop floor equipment or Manufacturing Execution Systems (MES) directly into OPM Process Execution.

In addition, these APIs serve as a central place to insert, update, or delete batch records in OPM from any source.

Following are some of the important characteristics for these APIs:

Code Reuse

You can maximize code reuse from all application development tools, including PL/SQL, Oracle Forms, and Oracle Reports.

Ease of Integration

You can integrate APIs into other applications and enabling technology, such as Oracle Workflow Server, Oracle Internet Commerce & Oracle WebSystem, and Oracle EDI Gateway.

Insulation from Changes

You can encapsulate the structure of schema to prevent changing schema structures from affecting other applications.

Consistent Behavior

You can hide Object logic specific to an application from other applications, and to ensure that this logic is correctly invoked by other applications and customers.

Robust Validation

You can validate all incoming information into Oracle Applications.

Input Data Sources

Flat File

Input data to the user wrapper function comes from a flat file source. This is processed by the wrapper and header information, passed as parameters, to the stored procedure in a synchronous mode. However, along with the standard parameters, the header information is passed as a PL/SQL table. In this mode, the calling function monitors the success or failure (return code) from the called procedure. It also provides an option to COMMIT work done by the procedure.

Batch File

Input data to the user wrapper function comes from a batch file. This is processed by the wrapper and header information passed, as parameters, to the stored procedure in an asynchronous mode. In this mode, the calling function does not monitor the success or failure of each individual record. The Oracle Message FND_ PUB_MSG functionality is used to audit the calls.

Online User Interface (UI)

Input data to the user wrapper function comes from a window or another user interface. This is processed by the UI and the details passed, as parameters, to the stored procedure in a synchronous mode. In this mode, the UI calling function monitors the success or failure (return code) from the called procedure. It also provides an option to COMMIT work done by the procedure.

Wrapper Function

Windows are generally used as wrapper functions.

The wrapper function is responsible for collating the details required as input parameters to the stored procedure and forwarding these in the call and monitoring the return code.

The stored procedure returns three possible return code:

- S for success
- E for error
- U for unknown or unexpected status
- N for item requiring a location
- V for inventory shortage exists
- I for incomplete manual transactions exist
- G for over allocation exists

Based on the return, the wrapper function searches the Oracle Messages File for the stored procedure to determine a COMMIT of the transaction or not.

Stored Procedure

The stored procedure is called with the appropriate parameters forwarded in a PL/SQL table format. The procedure validates each record from this table and then processes the appropriate functional logic as required. The procedure writes appropriate messages to the Oracle Messages table. These are informational as determined by the logic. These can be interrogated by the calling wrapper function through the GET MESSAGES functionality.

The stored procedure calls other validation procedures in the course of its execution; a modular approach has been adopted. Functions called by these procedures do not use IN/OUT parameters as these have been removed from the Oracle 8 coding standards.

On successful completion of the procedure, a success unit is in place that can be optionally COMMITTED. The decision as to whether a COMMIT is issued on successful completion is under the control of the calling code and deliberately outside the scope of the API procedures.

Major Features

In order to support the requirements mentioned in the "Basic Business Needs" topic, the new APIs support the following functionality:

Allocate Batch

The Allocate Batch API autoallocates a batch and all the phantom batches.

Allocate Line

The Allocate Line API autoallocates the material detail line.

Cancel Batch

The Cancel Batch API is a business object that nullifies a batch. The batch remains in the database for audit purposes, but it has a specific status of cancelled and any effect it has on inventory is reversed.

Close Batch

The Close Batch API is a business object that closes the batch. Closing a batch prevents further editing of a batch, and makes it available for use by Actual Costing.

Close Steps

The Close Step API is a business object that closes the batch step. Closing a batch step prevents further editing.

Complete Batch

The Complete Batch, or certify batch, API is a business object that completes the batch. Completing a batch indicates that the batch has been completed and the products and byproducts have been yielded. Completing a batch lets adjustments to ingredient, product, and byproduct quantities.

Complete Steps

The Complete Step, or certify batch step, API is a business object that completes a single batch step or multiple batch steps based on certain criteria. Completing a step is how output quantities are specified, or defaulted, for items yielded in the step. Resource usage for all activities is also calculated. If this is the final step, then completing it results in a batch completion when the GME:Step Control Batch Status profile option is set to Yes.

Convert FPO to Batches

The Convert FPO to Batches API is used to convert the firm planned order to one or more batches.

Create Batch

The Create Batch API is a business object that creates batches, lab batches, and Firm Planned Orders (FPOs) in OPM.

Create Phantom

The Create Phantom API creates phantom batches based on the validity rule passed.

Delete Batchstep Resource

An existing resource can be deleted for an activity of a step. A resource can be deleted for a pending step only.

Delete Material Detail Line

The Delete Material Detail Line API deletes the material line in the batch.

Delete Step

The Delete Step API deletes the step in the batch.

End Completed Resource Transaction

The End Completed Resource Transaction API is a business object that lets you create completed resource transactions for a specified end date.

Incremental Backflushing

The Incremental Backflushing, or partial certification, API is a business object that lets you incrementally record product yield and have ingredient usage backflushed.

Insert Incremental Completed Transaction

The Insert Incremental Completed Transaction API is a business object that lets you create completed resource transactions for the usage specified in addition to existing resource transactions.

Insert Batchstep Resource

A resource can be added for an activity of a step. A resource can be added for a step in pending, WIP, and completed step status only. If ASQC is on, then the resource cannot be added in WIP status since the process quantity cannot be inserted. Based on the step status and ASQC, all the input data is validated.

Insert Line Allocation

The Insert Line Allocation API lets you create pending or complete allocations for a given detail line in a given batch. The batch can be in a pending, WIP, or certified state and the material detail line can have any release type.

Insert Material Detail Line

The Insert Material Detail Line API is used to insert the material line into the batch.

Insert Step

The Insert Step API inserts the new step to the batch.

Insert Timed Resource Transaction

The Insert Timed Resource Transaction API is a business object that lets you create completed resource transactions for the specified start and end dates, and calculate the usage from in the resource usage unit of measure from this information.

Release Batch

The Release Batch API is a business object that converts pending batches to work in process (WIP) batches in OPM.

Release Steps

The Release Step API is a business object that can release a single batch step or multiple batch steps based on certain criteria. If the profile option GME:Step Control Batch Status is set to Yes, then it will also release the batch.

Reopen Batch

The Reopen Batch API changes the status of the batch from closed to completed.

Reopen Steps

The Reopen Steps API changes the status of the step from closed to completed.

Reroute Batch

The Reroute Batch API is a business object that reroutes a batch to a different recipe with the same formula. The batch or firm planned order is only rerouted in pending status.

Reschedule Batch

The Reschedule Batch API is a business object that reschedules a batch or firm planned order to a different date. The batch is only rescheduled while in pending or WIP status. The firm planned order is only rescheduled when in a pending state.

Reschedule Step

The Reschedule Step API reschedules the step and all subsequent steps.

Scale Batch

The Scale Batch API scales batches up or down, as well as all the phantom batches.

Start Completed Resource Transaction

The Start Completed Resource Transaction API is a business object that lets you create completed resource transactions for a specified start date. The end date is populated with the start date.

Theoretical Yield Batch

The Theoretical Yield Batch API calculates theoretical yield for the batch, and updates the quantities for the product lines.

Revert to WIP Batch

The Revert to WIP Batch, or uncertify batch, API is a business object that uncompletes the batch. Reverting a batch to WIP can set completed product and byproduct transactions to pending, and put the batch status back to WIP from completed.

Revert to WIP Steps

The Revert to WIP Steps, or uncertify batch steps, API is a business object that changes a batch step from completed to WIP.

Unrelease Batch

The Unrelease Batch API removes the completed allocations for the ingredient lines. Unreleasing a batch sets the batch status to pending.

Unrelease Steps

The Unrelease Step API is a business object that unreleases a batch step based on certain criteria. Unreleasing a step sets the step to pending status.

Update Actual Resource Usage

The Update Actual Resource Usage API is a business object that lets you create completed resource transactions for the usage specified, and deletes all existing transactions.

Update Batchset Resource

An existing resource can be updated for an activity of a step. A resource can be updated for a step in pending, WIP, and completed step status only. Based on the step status and ASQC, all the input data is validated before the resource in updated. Based on the step status, different fields of the resource can be updated.

Update Material Detail Line

The Update Material Detail Line API updates the material line in the batch.

Process Execution API Bill of Materials

The following are packages and files that are delivered with the OPM Process Execution APIs. These must be on your system for your interface to compile and link properly.

- GMEGAPIS.pls: GME_API_GRP
- GMEGTXNS.pls: GME_TRANS_ENGINE_GRP
- GMEMATLS.pls: GME_API_MATERIAL_DETAILS
- GMEPAPIS.pls: GME_API_PUB
- GMEUDBGS.pls: GME_DEBUG
- GMEUTXNS.pls: GME_TRANS_ENGINE_UTIL
- GMEVALBS.pls: GME_API_ALLOCATE_BATCH_PVT
- GMEVALLS.pls: GME_API_ALLOCATE_LINE_PVT
- GMEVCCBS.pls: GME_CANCEL_BATCH_PVT
- GMEVCCSS.pls: GME_CANCEL_STEP_PVT
- GMEVCFPS.pls: GME_API_CONVERT_FPO
- GMEVCLBS.pls: GME_API_CLOSE_BATCH
- GMEVCLSS.pls: GME_API_CLOSE_STEP
- GMEVCRBS.pls: GME_API_CREATE_BATCH
- GMEVCRSS.pls: GME_API_CREATE_STEP
- GMEVCTBS.pls: GME_API_CERTIFY_BATCH
- GMEVCTSS.pls: GME_API_CERTIFY_BATCH_STEP
- GMEVDBSS.pls: GME_API_DELETE_BATCH_STEP
- GMEVGBHS.pls: GME_BATCH_HEADER_DBL
- GMEVGBSS.pls: GME_BATCH_STEPS_DBL
- GMEVGHSS.pls: GME_BATCH_HISTORY_DBL
- GMEVGITS.pls: GME_INV_TXNS_GTMP_DBL
- GMEVGLBS.pls: GME_LAB_BATCH_LOTS_DBL
- GMEVGMDS.pls: GME_MATERIAL_DETAILS_DBL
- GMEVGRGS.pls: GME_RESOURCE_TXNS_GTMP_DBL
- GMEVGRTS.pls: GME_RESOURCE_TXNS_DBL
- GMEVGSAS.pls: GME_BATCH_STEP_ACTIVITIES_DBL
- GMEVGSDS.pls: GME_BATCH_STEP_DEPEND_DBL

- GMEVGSIS.pls: GME_BATCH_STEP_ITEMS_DBL
- GMEVGSOS.pls: GME_BATCH_SALES_ORDERS_DBL
- GMEVGSRS.pls: GME_BATCH_STEP_RESOURCES_DBL
- GMEVGSTS.pls: GME_BATCH_STEP_TRANSFERS_DBL
- GMEVINSS.pls: GME_API_INSERT_STEP
- GMEVPCBS.pls: GME_API_PARTIAL_CERT
- GMEVPHBS.pls: GME_API_PHANTOM
- GMEVRLBS.pls: GME_API_RELEASE_BATCH
- GMEVRLSS.pls: GME_API_RELEASE_BATCH_STEP
- GMEVROBS.pls: GME_REOPEN_BATCH_PVT
- GMEVROSS.pls: GME_REOPEN_STEP_PVT
- GMEVRRBS.pls: GME_API_REROUTE_BATCH
- GMEVRSBS.pls: GME_API_RESCHEDULE_BATCH
- GMEVRSSS.pls: GME_API_RESCHEDULE_BATCH_STEP
- GMEVRXNS.pls: GME_RESOURCE_ENGINE_PVT
- GMEVSCBS.pls: GME_API_SCALE_BATCH AUTHID CURRENT_USER
- GMEVTSTS.pls: GME_API_TEST
- GMEVTXNS.pls: GME_TRANS_ENGINE_PVT
- GMEVTXTS.pls: GME_TEXT_DBL
- GMEVUCBS.pls: GME_API_UNCERTIFY_BATCH
- GMEVUCSS.pls: GME_API_UNCERTIFY_BATCH_STEP
- GMEVURBS.pls: GME_API_UNRELEASE_BATCH
- GMEVURSS.pls: GME_API_UNRELEASE_STEP
- GMEVUSQS.pls: GME_API_UPDATE_STEP_QTY
- GMDPRDTS.pls: GMD_RECIPE_DATA_PUB
- GMDPCOMS.pls: GMD_COMMON_VAL
- GMDSTEPS.pls: GMD_AUTO_STEP_CALC
- GMDVSCLS.pls: GMD_COMMON_SCALE

- GMDPOPNS.pls: GMD_FETCH_OPRN
- GMDPRCFS.pls: GMD_RECIPE_FETCH_PUB
- GMDPVRFS.pls: GMD_FETCH_VALIDITY_RULES
- GMDPVRDS.pls: GMD_VAL_DATA_PUB
- GMDPRVRS.pls: GMD_VALIDITY_RULES
- GMDPRTVS.pls: GMDRTVAL_PUB
- GMAGMETS.pls: GMA_GME_TEXT_TBL_PKG

2 Process Execution API Usage

The Process Execution APIs are written in PL/SQL. To use these APIs, you must code your interface or wrapper. Your program is responsible for connecting to a database before calling an API function. You can write log files before calling and after returning from an API function. Each function returns an error code in the parameter x_return_status that indicates whether the API was successful or failed. The values are:

- S for success
- E for error
- U for unknown or unexpected status
- N for item requiring a location
- V for inventory shortage exists
- I for incomplete manual transactions exist
- G for over allocation exists

The topics discussed in this chapter are:

- Calling the API Interface Code
- API Wrapper Code - Example

Calling the API Interface Code

The following are part of a sample wrapper, and are used to test the API code. Wrappers are written as PL/SQL packages. Wrappers can be written for each API and call the APIs directly. The source of data for the wrapper comes from an ASCII flat file in this example. You can write a similar type of wrapper to call the API code.

These wrappers have the following parameters:

Standard Input Parameters

p_api_version	IN	NUMBER
p_validation_level	IN	NUMBER
p_init_msg_list	IN	BOOLEAN
p_commit	IN	BOOLEAN

Standard Output Parameters

x_message_count	OUT	NUMBER
x_message_list	OUT	VARCHAR2
x_return_status	OUT	VARCHAR2

API Call Hints

For performance improvement, NOCOPY hints have been added to the OUT parameters of the APIs. When an API has the same type of parameter defined as IN and OUT, you must pass in different variables. In addition, you must check the return status of the API (generally returned through x_return_status parameter) before looking at other OUT variables returned by the API. If the return status is not Success, then you must not use any of the OUT parameters passed back from the API.

For example, the Update_material_line API contains p_material_detail and x_ material_detail:

```
PROCEDURE update_material_line (
   p_api_version        IN         NUMBER := gme_api_pub.api_version
  ,p_validation_level   IN         NUMBER := gme_api_pub.max_errors
  ,p_init_msg_list      IN         BOOLEAN := FALSE
  ,p_commit             IN         BOOLEAN := FALSE
  ,x_message_count      OUT NOCOPY NUMBER
  ,x_message_list       OUT NOCOPY VARCHAR2
```

```
     ,x_return_status      OUT NOCOPY VARCHAR2
     ,p_material_detail    IN         gme_material_details%ROWTYPE
     ,p_values_tab         IN         gme_api_pub.field_values_tab
     ,p_scale_phantom      IN         BOOLEAN := FALSE
     ,x_material_detail    OUT NOCOPY gme_material_details%ROWTYPE);
```

Therefore, the call can be set up to read:

```
gme_api_main.update_material_line(p_api_version      => p_api_version
                                 ,p_validation_level => p_validation_level
                                 ,p_init_msg_list    => FALSE
                                 ,p_commit           => p_commit
                                 ,x_message_count    => x_message_count
                                 ,x_message_list     => x_message_list
                                 ,x_return_status    => x_return_status
                                 ,p_material_detail  => l_material_detail
                                 ,p_values_tab       => p_values_tab
                                 ,p_scale_phantom    => p_scale_phantom
                                 ,x_material_detail  => l_material_detail);
```

In this example, p_material_detail and x_material_detail both have the variable l_ material_detail. This gives an incorrect result because both the parameters cannot have the same variable.

You must set the call up so that p_material_detail and x_material_detail have different variables:

```
gme_api_main.update_material_line(p_api_version      => p_api_version
                                 ,p_validation_level => p_validation_level
                                 ,p_init_msg_list    => FALSE
                                 ,p_commit           => p_commit
                                 ,x_message_count    => x_message_count
                                 ,x_message_list     => x_message_list
                                 ,x_return_status    => x_return_status
                                 ,p_material_detail  => l_material_detail_in
                                 ,p_values_tab       => p_values_tab
                                 ,p_scale_phantom    => p_scale_phantom
                                 ,x_material_detail  => l_material_detail_out);
```

API Wrapper Code - Example

```
--+==========================================================================+
--| PROCEDURE NAME      |
--| Create_Batch      |
--|       |
--| TYPE      |
--| Public      |
--|       |
--| USAGE      |
--| Create_Batch      |
--|       |
--| DESCRIPTION      |
--| This is a PL/SQL wrapper function to call the Create Batch API.           |
--|                                                                           |
--| REQUIREMENTS                                                              |
--|   This wrapper assumes that the user has initialized the application     |
--|   user variables. To do this the procedure fnd_global.apps_initialize    |
--|   needs to be invoked with the appropriate user id and responsibility    |
--|                                                                           |
--|   The user has to supply the following required values                   |
--|     p_batch_type              0 - Batch, 10 - Firm Planned Order         |
--|     p_orgn_code               Plant in which the batch is created        |
--|     p_creation_mode           PRODUCT, RECIPE, OUTPUT or INPUT           |
--|     p_batch_size              The size of the batch to be created        |
--|     p_batch_size_uom          The Unit of measure code of the batch size|
--|     p_recipe_validity_rule_id  The recipe validity rule ID with which the|
--|                               batch has to be created. This field could |
--|                               be omitted if any of the following combi- |
--|                               nations are provided                       |
--|                               p_recipe_id or                             |
--|                               p_recipe_no, p_recipe_version or           |
--|                               p_product_id (Item_Id) or                  |
--|                               p_product_no (Item_No)                     |
--|     p_batch_no                This is required if the plant is set for  |
--|                               manual document ordering                   |
--|     Out Variables                                                        |
--+==========================================================================+

PROCEDURE create_batch
    ( p_api_version            IN      NUMBER   DEFAULT gme_api_pub.api_version
     ,p_validation_level       IN      NUMBER   DEFAULT gme_api_pub.max_errors
     ,p_init_msg_list          IN      BOOLEAN  DEFAULT FALSE
     ,p_commit                 IN      BOOLEAN  DEFAULT FALSE

     ,P_batch_type             IN      NUMBER
     ,p_orgn_code              IN      VARCHAR2
     ,p_creation_mode          IN      VARCHAR2
     ,p_batch_size             IN      NUMBER
     ,p_batch_size_uom         IN      VARCHAR2
```

```
      ,p_plan_start_date        IN      DATE     DEFAULT SYSDATE
      ,p_plan_cmplt_date        IN      DATE     DEFAULT SYSDATE
      ,p_due_date               IN      DATE     DEFAULT SYSDATE
      ,p_update_inventory_ind   IN      VARCHAR2 DEFAULT 'Y'

      ,p_recipe_validity_rule_id IN     NUMBER   DEFAULT NULL

      ,p_recipe_id              IN      NUMBER   DEFAULT NULL
      ,p_recipe_no              IN      VARCHAR2 DEFAULT NULL
      ,p_recipe_version          IN      NUMBER   DEFAULT NULL
      ,p_product_no              IN      VARCHAR2 DEFAULT NULL
      ,p_product_id              IN      NUMBER   DEFAULT NULL
      ,p_batch_no                IN      VARCHAR2 DEFAULT NULL

      ,p_ignore_qty_below_cap   IN      BOOLEAN  DEFAULT TRUE
      ,p_ignore_shortages       IN      BOOLEAN  DEFAULT TRUE

      ,x_batch_header           OUT      gme_batch_header%ROWTYPE
      ,x_unallocated_material   OUT      gme_api_pub.unallocated_materials_tab

      ,x_message_count          OUT      NUMBER
      ,x_message_list           OUT      VARCHAR2
      ,x_return_status          OUT      VARCHAR2 ) IS

   l_batch_headerGME_BATCH_HEADER%ROWTYPE;
   l_msg_index_outNUMBER;
 BEGIN
   /* Enable The Buffer */
   DBMS_OUTPUT.ENABLE(1000000);

   /* Let us build the batch row type variable with the input values */
   l_batch_header.plant_code := p_orgn_code;
   l_batch_header.batch_type := p_batch_type;
   l_batch_header.batch_no:= p_batch_no;
   l_batch_header.plan_start_date := p_plan_start_date;
   l_batch_header.plan_cmplt_date := p_plan_cmplt_date;
   l_batch_header.due_date := p_due_date;
   l_batch_header.update_inventory_ind := p_update_inventory_ind;

   IF p_recipe_validity_rule_id IS NOT NULL THEN
     l_batch_header.recipe_validity_rule_id := p_recipe_validity_rule_id;
   END IF;

   gme_api_pub.create_batch(
       p_api_version            =>  p_api_version
      ,p_validation_level       =>  p_validation_level
      ,p_init_msg_list          =>  p_init_msg_list
      ,p_commit                 =>  p_commit

      ,x_message_count          =>  x_message_count
      ,x_message_list           =>  x_message_list
      ,x_return_status          =>  x_return_status

      ,p_batch_header           =>  l_batch_header
      ,x_batch_header           =>  x_batch_header

      ,p_batch_size             =>  p_batch_size
      ,p_batch_size_uom         =>  p_batch_size_uom
      ,p_creation_mode          =>  p_creation_mode
      ,p_recipe_id       =>  p_recipe_id
      ,p_recipe_no       =>  p_recipe_no
      ,p_recipe_version       =>  p_recipe_version
      ,p_product_no       =>  p_product_no
      ,p_product_id       =>  p_product_id

      ,p_ignore_qty_below_cap   =>  p_ignore_qty_below_cap
      ,p_ignore_shortages       =>  p_ignore_shortages

      ,x_unallocated_material   =>  x_unallocated_material );

   IF x_return_status <> FND_API.g_ret_sts_success THEN
     IF X_message_count = 1 THEN
       DBMS_OUTPUT.PUT_LINE('Error:'||X_message_list);
     ELSE
       FOR i IN 1..x_message_count LOOP
         FND_MSG_PUB.get (p_msg_index     => i
                         ,p_data          => X_message_list
                         ,p_msg_index_out => l_msg_index_out);
         DBMS_OUTPUT.PUT_LINE('Error:'||X_message_list);
       END LOOP;
     END IF;
   ELSE
     DBMS_OUTPUT.PUT_LINE(' A new batch '||x_batch_header.batch_no||' has been
 created ');
     IF x_unallocated_material.count > 0 THEN
       DBMS_OUTPUT.PUT_LINE(' Items failing auto allocation:');
       FOR i IN 1..x_unallocated_material.COUNT LOOP
         DBMS_OUTPUT.PUT_LINE('  Line Type:'||x_unallocated_material(i).line_
 type||' Line No:'||x_unallocated_material(i).line_no||
                               ' Item:'||x_unallocated_material(i).item_no||'
 Allocated:'||x_unallocated_material(i).alloc_qty||
                               ' Unalloc:'||x_unallocated_material(i).unalloc_
 qty||' UOM:'||x_unallocated_material(i).alloc_uom);
       END LOOP;
     END IF;
   END IF;
 EXCEPTION
   WHEN OTHERS THEN
     x_return_status := FND_API.g_ret_sts_unexp_error;
     x_message_count := 1;
     x_message_list  := SQLERRM;
     dbms_output.put_line('Error '||TO_CHAR(SQLCODE)||': '||SQLERRM);
 END Create_Batch;
```

3 Technical Overview

The public Process Execution APIs perform all validations necessary on input data supplied in order to prevent the flow of invalid data into OPM. If any validation errors occur, then that particular row is skipped and the process continues to the next record. If the insert fails, then none of the detail records for the item in process are inserted. After finishing validations on input data, the public API performs the required function by calling the necessary routines.

The topics discussed in this chapter are:

- Structure for Process Execution Public APIs
- Standard Parameters

Structure for Process Execution Public APIs

According to API standards, the following are the files, packages, and procedures for the public APIs.

Object Type	**Name**
Package Specification File	GMEPAPIS.pls
Package Body File	GMEPAPIB.pls
Package	gme_api_pub
Procedure - Allocate Batch	allocate_batch
Procedure - Allocate Line	allocate_line
Procedure - Cancel Batch	cancel_batch
Procedure - Close Batch	close_batch
Procedure - Close Batch Steps	close_step
Procedure - Complete Batch	certify_batch
Procedure - Complete Batch Steps	certify_batch_step
Procedure - Convert FPO to Batches	convert_fpo
Procedure - Create Batch	create_batch
Procedure - Create Phantom	create_phantom
Procedure - Delete Batchset Resource	delete_batchstepresource
Procedure - Delete Material Line Detail	delete_material_line
Procedure - Delete Step	delete_step
Procedure - Incremental Backflushing	partial_cert_batch
Procedure - End Completed Resource Transaction	end_cmplt_actual_rsc_txn
Procedure - Insert Batchstep Resource	insert_batchstepresource
Procedure - Insert Incremental Completed Transaction	insert_incr_actual_rsrc_txn
Procedure - Insert Line Allocation	insert_allocation
Procedure - Insert Material Line Detail	insert_material_line
Procedure - Insert Step	insert_step

Object Type	Name
Procedure - Insert Timed Resource Transaction	insert_timed_actual_rsrc_txn
Procedure - Release Batch	release_batch
Procedure - Release Batch Steps	release_step
Procedure - Reopen Batch	reopen_batch
Procedure - Reopen Batch Steps	reopen_step
Procedure - Reroute Batch	reroute_batch
Procedure - Reschedule Batch	reschedule_batch
Procedure - Reschedule Batch Step	reschedule_step
Procedure - Scale Batch	scale_batch
Procedure - Start Completed Resource Transaction	start_cmplt_actual_rsrc_txn
Procedure - Theoretical Yield Batch	theoretical_yield_batch
Procedure - Revert to WIP Batch	uncertify_batch
Procedure - Revert to WIP Batch Steps	uncertify_batch_step
Procedure - Unrelease Batch	unrelease_batch
Procedure - Unrelease Batch Steps	unrelease_step
Procedure - Update Actual Resource Usage	update_actual_resource_usage
Procedure - Update Batchset Resource	update_batchstepresource
Procedure - Update Material Detail Line	update_material_line

Standard Parameters

API standard parameters are a collection of parameters that are common to most APIs. The following paragraphs explain the standard parameters used in APIs and their interpretation.

Some of the standard parameters apply to all APIs regardless of the nature of the business function they perform. For example, p_api_version and x_return_status are included in all APIs.

Some parameters are applicable for certain types of APIs and not applicable for other types. For example, p_commit is applicable for APIs that change the database state, and not applicable for read APIs.

Standard parameters are included in all APIs whenever applicable.

Standard IN parameters:

- p_api_version
- p_init_msg_list
- p_commit
- p_validation_level

Standard OUT parameters:

- x_return_status
- x_msg_count
- x_msg_data

Parameter	Type	IN/OUT	Required	Validation
p_api_ version	varchar2	IN	Y	Validates version compatibility. The version sent by the calling function is compared to the internal version of the API and an unexpected error (U) is generated if these do not match.

Parameter	Type	IN/OUT	Required	Validation
p_init_msg_ list	boolean	IN	N	Used to specify whether the message list is initialized on entry to the API. It is an optional parameter, and if not supplied, defaults to FND_API.G_ FALSE which means that the API does not initialize the message list. If multiple APIs are called in same session, you must pass this as TRUE, otherwise messages accumulate on the stack.
p_commit	boolean	IN	N	Used to specify whether the API commits its work before returning to the calling function. If not supplied, then it defaults to FALSE. You should ensure that the save_batch procedure is called before peforming any commits manually. You must ensure that the save_batch procedure is called before performing any commits manually (if p_commit parameter is passed as FALSE).
x_return_ status	varchar2	OUT	N	Specifies whether the API was successful or failed. Valid values are S for successful, E for failed due to expected error, or U for failed due to unexpected error.
x_msg_ count	number	OUT	N	Specifies the number of messages added to message list.
x_msg_data	varchar2	OUT	N	Returns the messages in an encoded format. These messages are processed by the standard message functions as defined in the Business Object API Coding Standards Document.

Value-ID Conversion

IDs are usually used to represent primary and foreign entity keys, and for internal processing of attributes. They are not meaningful and are hidden. Besides IDs, attributes have values that represent them. Those values are meaningful and are used for display purposes. In general, APIs operate only on IDs.

For example, an item is represented by an ID, the number column item_id. This ID is its primary key and is used for all internal processing of the item. Besides this ID, an item is represented by a value, the varchar2 column item_no. This value is displayed when you choose an item. Therefore, an item can be identified by either its ID or value, in this case item_no.

The following set of rules are for the conversion process:

- Either ID or value, or both can be passed to an API. But, when both values are passed, ID based parameters take precedence over value based parameters. For example, if both parameters are passed, the value based parameter is ignored and the ID based parameter is used.
- When both the value and ID of an attribute are passed to an API, a message informs the API caller that some of the input has been ignored.
- This message is not an error message. The API continues with its regular processing.
- Each value has to resolve into one ID. Failure to resolve a value into an ID is an error and is associated with an error message. The API aborts processing and returns with a return status of error.

4 Business Objects

Allocate Batch

This procedure autoallocates ingredients in a batch and all phantom batches.

Following is the definition of Allocate Batch:

Parameter	Required	In/Out	Description
p_allocate_type	N	IN	For internal use. Always set to 9.
p_release_type	N	IN	For internal use. Always set to 9.
p_batch_header	Y	IN	The batch header row to identify the batch.
x_batch_header	Y	OUT	The batch header that is returned, with all the data.
p_del_exist_alloc	N	IN	Delete existing allocations before autoallocating. Default value is False.
x_unallocated_material	Y	OUT	Table of materials, unallocated items exist.

Parameter - p_batch_header (IN)

This is a row type parameter that identifies the batch header. The following table explains the required columns of the row:

Parameter	Description
batch_id	Unique identifier for batches.
batch_no	Batch number for the batch. Batch_no, plant_code, and batch_type are required if batch_id is not set.

Parameter	Description
plant_code	Organization for which the batch was created. Batch_no, plant_code, and batch_type are required if batch_id is not set.
batch_type	Type of batch. Values include 0=batch, 10=firm planned order. Batch_no, plant_code, and batch_type are required if batch_id is not set.

Parameter - x_batch_header (OUT)

This is a row type parameter that returns the populated batch header for the allocated batch.

Parameter - x_unallocated_material (OUT)

This is a table type parameter that holds the information of the unallocated material, inventory shortage, or incomplete manual allocations. The following table explains how these IDs are returned:

Parameter	Description
batch_id	Unique identifier for batches to firm planned order or batch.
batch_no	Batch or firm planned order number.
material_detail_id	Unique identifier for a batch or firm planned order line.
line_type	Valid values are -1 = Ingredient, 1 = Product; The product on Line 1 is the primary product, 2 = Byproduct
line_no	Sequential line number for each line type in a batch or firm planned order.
item_id	FK to the item that is a product, ingredient, or byproduct.
item_no	Item Number.
alloc_qty	Quantity that is allocated.
unalloc_qty	The unallocated quantity.
alloc_uom	Unit of measure for ALLOC_QTY and UNALLOC_QTY also the material lines ITEM_UM.

Allocate Line

The Allocate Line API refers to specifying the lots and locations of items used for a batch. Allocation can be performed automatically by OPM or manually.

This API provides a way to autoallocate a particular ingredient line of batch.

Following is the definition of Allocate Line:

Parameter	Required	In/Out	Description
p_material_details	Y	IN	The material detail row to identify the material.
p_allocate_type	N	IN	For internal use. Always set to 9.
p_del_exist_alloc	N	IN	Delete existing allocations before autoallocating. Default value is False.

Parameter - p_material_details (IN)

This is a row type parameter that identifies the material detail. The following table explains the required columns of the row:

Parameter	Description
material_detail_id	Unique identifier for a batch or firm planned order line.

Cancel Batch

In OPM, batches and firm planned orders (FPOs) are created whenever a production run for one or more products of a given recipe need to be manufactured. After creation they are progressed through the various stages of the manufacturing cycle:

- Firm planned orders are converted to batches of the required size, number, or size and number.
- Batches are progressed from pending through WIP, completed and closed.

This is the normal business sequence, but there are times when it is necessary to go back a stage in the cycle.

Once a firm planned order has been converted to batches, it is not possible to unconvert the batches to a firm planned order. However, it is possible to cancel a pending batch so that it is annulled.

Firm planned orders can also be cancelled in an identical manner to pending batches. If the firm planned order is partially converted, for example a 2000kg firm planned order has so far only been converted to batches totalling 1000kg, then the unconverted residue is cancelled. The batches already created by partially converting the firm planned order remains untouched.

If the batch being cancelled contains steps, then this API cancels all steps in the batch. There is no publicly called Cancel Step API.

If a batch or firm planned order has been cancelled, then there is no way to uncancel it.

Following is the definition of Cancel Batch:

Parameter	Required	In/Out	Description
p_batch_header	Y	IN	The batch header row to identify the batch.
x_batch_header	Y	OUT	The batch header that is returned, with all the data.

Parameter - p_batch_header (IN)

This is a row type parameter that identifies the batch header. The following table explains the required columns of the row:

Parameter	Description
batch_id	Unique identifier for batches to firm planned order or batch.

Parameter	Description
batch_no	Batch number for the batch. Batch_no, plant_code, and batch_type are required if batch_id is not set.
plant_code	Organization for which the batch was created. Batch_no, plant_code, and batch_type are required if batch_id is not set.
batch_type	Type of batch. Values include 0=batch, 10=firm planned order. Batch_no, plant_code, and batch_type are required if batch_id is not set.

Parameter - x_batch_header (OUT)

This is a row type parameter that returns the populated batch header for the cancelled batch.

Close Batch

The Close Batch API is a business object that can close the batch. Closing a batch prevents further editing.

Only a completed batch can be closed. All steps are closed at that time.

Following is the definition of Close Batch:

Parameter	Required	In/Out	Description
p_batch_header	Y	IN	The batch header row to identify the batch.
x_batch_header	Y	OUT	The batch header that is returned, with all the data.

Parameter - p_batch_header (IN)

This is a row type parameter that identifies the batch header. The following table explains the required columns of the row:

Parameter	Description
batch_id	Unique identifier for batches.
batch_no	Batch number for the batch. Batch_no, plant_code, and batch_type are required if batch_id is not set.
plant_code	Organization for which the batch was created. Batch_no, plant_code, and batch_type are required if batch_id is not set.
batch_type	Type of batch. Values include 0=batch, 10=firm planned order. Batch_no, plant_code, and batch_type are required if batch_id is not set.
batch_close_date	The date and time the batch was closed.

Parameter - x_batch_header (OUT)

This is a row type parameter that returns the populated batch header for the closed batch.

Close Steps

The Close Step API is a business object that can close the batch step. Closing a batch step prevents further editing.

When you close a batch step, any items associated with that step, with consumption or yield type of Automatic by Step, are not editable. Only a completed batch steps can be closed.

The immediately prior dependent step must have a status of closed to close the batch step. The batch step row that is passed in must contain sufficient information to identify the step, this can be batch_id and batchstepno or batchstep_id.

Following is the definition of Close Step:

Parameter	Required	In/Out	Description
p_batch_step	Y	IN	The batch step row to identify the batch step.
x_batch_step	Y	OUT	The batch step that is returned, with all the data.
p_delete_pending	Y	IN	Deletes the pending allocations for the material lines associated with the step.

Parameter - p_batch_step (IN)

This is a row type parameter that identifies the batch step. The following table explains the required columns of the row:

Parameter	Description
batchstep_id	Unique identifier for batch steps.
batch_id	ID of the batch. Batch_id and batchstep_no are required if batchstep_ id is not set.
batchstep_no	Step number. Batch_id and batchstep_no are required if batchstep_id is not set.
step_close_date	Entered at the time the step is closed. Can be entered by you or current system date.

Parameter - x_batch_step (OUT)

This is a row type parameter that returns the populated batch step for the closed step.

Complete Batch

The complete batch API is a business object that can complete the batch. Completing a batch indicates that the batch has been completed and the products and byproducts have been yielded. Completed batches still let adjustments to ingredient, product, and byproduct quantities.

If any of the ingredients, products, or byproducts are not allocated, then the complete batch routine returns back the unallocated material lines in the unallocated materials structure.

Only batches in pending or WIP status can be certified. All steps are also certified at this time. The batch header that is passed in must contain sufficient information to identify the batch (firm planned orders are not eligible for certifying).

Following is the definition of Complete Batch:

Parameter	Required	In/Out	Description
p_del_incomplete_ manual	N	IN	Delete incomplete manual transactions. Default value is False.
p_ignore_shortages	N	IN	If set to TRUE, it does not return x_ unallocated_material. This only takes effect if the GMI:Allow Negative Inventory profile options is set to 2, Warning. Default value is False.
p_batch_header	Y	IN	The batch header row to identify the batch.
x_batch_header	Y	OUT	The batch header that is returned, with all the data.
x_unallocated_material	Y	OUT	Table of materials, if inventory shortage exists, or incomplete manual transactions exist, or unallocated items exist.

Parameter - p_batch_header (IN)

This is a row type parameter that identifies the batch header. The following table explains the required columns of the row:

Parameter	Description
batch_id	Unique identifier for batches to firm planned order or batch.
batch_no	Batch number for the batch. Batch_no, plant_code, and batch_type are required if batch_id is not set.

Parameter	Description
plant_code	Organization for which the batch was created. Batch_no, plant_code, and batch_type are required if batch_id is not set.
batch_type	Type of batch. Values include 0=batch, 10=firm planned order. Batch_no, plant_code, and batch_type are required if batch_id is not set.
actual_start_date	Actual start date and time for batch. Entered in complete batch only if the batch was pending. Defaults to current system date.
actual_cmplt_date	Actual completion date. Entered at the time the batch is certified completed. Can be entered by you or current system date.

Parameter - x_batch_header (OUT)

This is a row type parameter that returns the populated batch header for the completed batch.

Parameter - x_unallocated_material (OUT)

This is a table type parameter that holds the information of the unallocated material, inventory shortage, or incomplete manual allocations. The following table explains how these IDs are returned:

Parameter	Description
batch_id	Unique identifier for batches to firm planned order or batch.
batch_no	Batch or firm planned order number.
material_detail_id	Unique identifier for a batch or firm planned order line.
line_type	Valid values are -1 = Ingredient, 1 = Product; The product on Line 1 is the primary product, 2 = Byproduct
line_no	Sequential line number for each line type in a batch or firm planned order.
item_id	FK to the item that is a product, ingredient, or byproduct.
item_no	Item Number.
alloc_qty	Quantity that is allocated.
unalloc_qty	Quantity that is unallocated.
alloc_uom	Unit of measure for ALLOC_QTY and UNALLOC_QTY also the material lines ITEM_UM.

Complete Steps

The complete batch step API is a business object that can complete a single batch step or multiple batch steps based on certain criteria. Completing a step is the way that output quantities are specified (or defaulted) for items yielded in the step. Actual resource usage for all the resources is also calculated. The API operates on steps that have a status of WIP or pending. If this is the final step, then completing it completes the batch if the GME:Step Controls Batch Status profile options is set to Yes.

If any of the ingredients, products, or byproducts associated with the step that have a consumption or yield type of Automatic by Step are not allocated, then the complete batch step routine returns the unallocated material lines, or incomplete manual in the unallocated materials structure.

Following is the definition of Complete Step:

Parameter	Required	In/Out	Description
p_del_incomplete_ manual	N	IN	Delete incomplete manual transactions. Default value is False.
p_ignore_shortages	N	IN	If set to TRUE, it does not return x_ unallocated_material. This only takes effect if the GMI:Allow Negative Inventory profile options is set to 2, Warning. Default value is False.
p_batch_step	Y	IN	The batch step row to identify the step.
p_override_quality	N	IN	If you have access to the function GMESTPED_COMPLETE_WITHOUT_ QC_F, this parameter enables you to override the quality status and complete the step, even though quality has not been approved. This is done by passing the value TRUE. The default value is FALSE.
x_batch_step	Y	OUT	The batch step that is returned, with all the data.
x_unallocated_material	Y	OUT	Table of materials, if inventory shortage exists, or incomplete manual transactions exist, or unallocated items exist.

Parameter - p_batch_step (IN)

This is a row type parameter that identifies the batch step. The following table explains the required columns of the row:

Parameter	Description
batch_id	ID of the batch. Batch_id and batchstep_no are required if batchstep_ id is not set.
batchstep_no	Step number. Batch_id and batchstep_no are required if batchstep_id is not set.
batchstep_id	Unique identifier for batch steps.
actual_start_date	Actual start date and time for batch step. Entered in complete step only if the step was pending. Defaults to current system date.
actual_cmplt_date	Actual completion date. Entered at the time the batch is completed. Can be entered by you or current system date.

Parameter - x_batch_step (OUT)

This is a row type parameter that returns the populated batch step for the completed step.

Parameter - x_unallocated_material (OUT)

This is a table type parameter that holds the information of the unallocated material, inventory shortage, or incomplete manual allocations. The following table explains how these IDs are returned:

Parameter	Description
batch_id	Unique identifier for batches.
batch_no	Batch or firm planned order number.
material_detail_id	Unique identifier for a batch or firm planned order line.
line_type	Valid values are -1 = Ingredient, 1 = Product; The product on Line 1 is the primary product, 2 = Byproduct
line_no	Sequential line number for each line type in a batch or firm planned order.
item_id	FK to the item that is a product, ingredient, or byproduct.
item_no	Item Number.
alloc_qty	Quantity that is allocated.
unalloc_qty	Quantity that was unallocated.
alloc_uom	Unit of measure for ALLOC_QTY and UNALLOC_QTY also the material lines ITEM_UM.

Convert FPO to Batches

The Convert FPO to Batches API is used to convert the firm planned order to a batch.

When you partially convert a firm planned order to a batch, the firm planned order is scaled down to contain only the remaining quantity.

Following is the definition of Convert FPO to Batches:

Parameter	Required	In/Out	Description
p_enforce_vldt_check	Y	IN	For internal use. Always set to FALSE.
p_batch_header	Y	IN	The batch header row to identify the batch.
x_batch_header	Y	OUT	The batch header that is returned, with all the data.
p_batch_size	Y	IN	The size of the batch to be created.
p_num_batches	Y	IN	Number of batches to be created. This must be greater than or equal to one.
p_validity_rule_id	Y	IN	If use_for_all is true, or if you are only converting one batch, then this parameter is required.
p_validity_rule_tab	Y	IN	This is used only if a firm planned order is converted to multiple batches with different validity rules.
p_leadtime	Y	IN	The duration of the batch.
p_batch_offset	Y	IN	If there are multiple batches, then this is the offset time between the batches.
p_offset_type	Y	IN	The offset type. Valid values are 0 - start to start, 1 - finish to start.
p_plan_start_date	Y	IN	The start date of the batch. Defaults to system date, but can be overridden.
p_plan_cmplt_date	Y	IN	The completion date of the batch.
p_use_for_all	N	IN	Use the same validity rule for all batches. Default value is True.

Parameter - p_batch_header (IN)

This is a row type parameter that identifies the batch header. The following table explains the required columns of the row:

Parameter	Description
batch_id	Unique identifier for batches to firm planned order or batch.
batch_no	Batch number for the batch. Batch_no, plant_code, and batch_type are required if batch_id is not set.
plant_code	Organization for which the batch was created. Batch_no, plant_code, and batch_type are required if batch_id is not set.
batch_type	Type of batch. Values include 0=batch, 10=firm planned order. Batch_ no, plant_code, and batch_type are required if batch_id is not set.

Parameter - x_batch_header (OUT)

This is a row type parameter that returns the populated batch header for the completed batch.

Parameter - p_validity_rule_tab (IN)

This is a table type parameter that holds the information of the unallocated material, inventory shortage, or incomplete manual allocations. The following table explains how these IDs are returned:

Parameter	Description
validity_rule_id	Validity Rule ID

Create Batch

The Create Batch API is a publicly callable stored procedure that creates batches, lab batches, and firm planned orders in OPM.

Batch creation is affected by calling the API with a set of parameters that indicates the following:

- Which recipe to use for the batch (lab batch or firm planned order).
- How it can be created (total input, total output).
- The batch quantity or unit of measure.
- An indication of whether the batch creation can proceed in the presence of inventory shortages for those ingredients that are allocated automatically.

If the batch, lab batch, or firm planned order is created successfully, then the API returns with a status of S. If errors occur, then the return status is E for errors that are normal but prevent the batch from being created, and U for errors that are unexpected and prevent the batch from being created, such as database errors.

On successful creation, a fully populated batch header is returned to the caller with any values that were not filled in, so that the caller has immediate knowledge of the surrogates allocated.

Following is the definition of Create Batch:

Parameter	Required	In/Out	Description
p_batch_header	Y	IN	The batch header row to identify the batch.
x_batch_header	Y	OUT	The batch header that is returned, with all the data.
p_batch_size	Y	IN	Batch size equalling total input, total output, or product quantity.
p_batch_size_uom	Y	IN	Unit of measure for batch size.
p_creation_mode	Y	IN	How the batch is created. Valid values are RECIPE, PRODUCT, TOTAL_OUTPUT, and TOTAL_INPUT.
p_ignore_shortages	Y	IN	If set to TRUE, it does not return x_ unallocated_material. This only takes effect if the GMI:Allow Negative Inventory profile options is set to 2, Warning.

Parameter	Required	In/Out	Description
p_recipe_id	N	IN	The recipe ID for the batch to be created. Default value is Null.
p_recipe_no	N	IN	The recipe number and recipe version for the batch to be created. Default value is Null.
p_recipe_version	N	IN	The version of the recipe for the batch to be created. Default value is Null.
p_product_no	N	IN	Item number for the batch to be created. Default value is Null.
p_product_id	N	IN	The product ID for the batch to be created.
p_ignore_qty_below_cap	N	IN	Indicates whether the batch is to be created or not, when resource quantity goes down below the minimum capacity of the resource. Default value is True.
x_unallocated_material	Y	OUT	Table of materials, if inventory shortage exists, or unallocated items exist.

Parameter - p_batch_header (IN)

This is a row type parameter that identifies the batch header. The following table explains the required columns of the row:

Parameter	Description
batch_no	Batch number for the batch if it is being created in a plant with manual document ordering. Batch_no, plant_code, and batch_type are required.
plant_code	Organization for which the batch was created. Batch_no, plant_code, and batch_type are required if batch_id is not set.
batch_type	Type of batch. Values include 0=batch, 10=firm planned order. Batch_no, plant_code, and batch_type are required if batch_id is not set.
plan_start_date	Planned start date and time for batch. Defaults to system date, but can be overridden.
plan_cmplt_date	Planned completion date. If a routing is used, or if production rules are set up, these are used to calculate the planned completion.
due_date	Required batch completion date and time.

Parameter	Description
recipe_validity_rule_id	Surrogate key to the Recipe Validity Rule Id the batch or FPO is based on.
wip_whse_code	Warehouse used to cost production activity.

Parameter - x_batch_header (OUT)

This is a row type parameter that returns the populated batch header for the created batch.

Parameter - x_unallocated_material (OUT)

This is a table type parameter that holds the information of the unallocated material or inventory shortage. The following table explains how these IDs are returned:

Parameter	Description
batch_id	Unique identifier for batches to firm planned order or batch.
batch_no	Batch or firm planned order number.
material_detail_id	Unique identifier for a batch or firm planned order line.
line_type	Valid values are -1 = Ingredient, 1 = Product; The product on Line 1 is the primary product, 2 = Byproduct
line_no	Sequential line number for each line type in a batch or firm planned order.
item_id	FK to the item that is a product, ingredient, or byproduct.
item_no	Item Number.
alloc_qty	Quantity that is allocated.
unalloc_qty	Quantity that was unallocated.
alloc_uom	Unit of measure for ALLOC_QTY and UNALLOC_QTY also the material lines ITEM_UM.

Create Phantom

The Create Phantom API creates phantom batches based on the validity rule passed.

Following is the definition of Create Phantom:

Parameter	Required	In/Out	Description
p_material_details	Y	IN	The material detail row to identify the material.
x_material_details	Y	OUT	The material detail that is returned, with all the data.
p_validity_rule_id	Y	IN	Validity rule to use for creating a phantom batch.
p_ignore_shortages	Y	IN	If set to TRUE, it does not return x_ unallocated_material. This only takes effect if the GMI:Allow Negative Inventory profile options is set to 2, Warning.
x_unallocated_material	Y	OUT	Table of materials, if inventory shortage exists, or incomplete manual transactions exist, or unallocated items exist.
p_batch_no	N	IN	If the organization is set for manual document ordering, then you must pass in this parameter. If the organization is set to be automatic document numbering, then this parameter is ignored.

Parameter - p_material_details (IN)

This is a row type parameter that identifies the material details. The following table explains the required columns of the row:

Parameter	Description
batch_id	To identify the batch. Batch_id, line_type and line_no are used if material_detail_id is not set.
line_type	Valid values are -1 = Ingredient, 1 = Product; The product on Line 1 is the primary product, 2 = Byproduct
line_no	Sequential line number for each line type in a batch or firm planned order.
material_detail_id	Unique identifier for a batch or firm planned order line.

Parameter - x_material_details (OUT)

This is a row type parameter that returns the populated material details for the created phantom.

Parameter - x_unallocated_material (OUT)

This is a table type parameter that holds the information of the unallocated material or inventory shortage. The following table explains how these IDs are returned:

Parameter	Description
batch_id	Unique identifier for batches to firm planned order or batch.
batch_no	Batch or firm planned order number.
material_detail_id	Unique identifier for a batch or firm planned order line.
line_type	Valid values are -1 = Ingredient, 1 = Product; The product on Line 1 is the primary product, 2 = Byproduct
line_no	Sequential line number for each line type in a batch or firm planned order.
item_id	FK to the item that is a product, ingredient, or byproduct.
item_no	Item Number.
alloc_qty	Quantity that is allocated.
unalloc_qty	Quantity that was unallocated.
alloc_uom	Unit of measure for ALLOC_QTY and UNALLOC_QTY also the material lines ITEM_UM.

Delete Batchstep Resource

An existing resource can be deleted for an activity of a step. A resource can be deleted for a pending step only.

Following is the definition of Delete Batchstep Resource:

Parameter	Required	In/Out	Description
p_batch_no	N	IN	Batch number.
p_plant_code	N	IN	Organization code.
p_resource	N	IN	Resource name.
p_batchstep_no	N	IN	Step number.
p_activity	N	IN	Activity name.
p_batchstep_resource_ id	Y	IN	Batchstep resource ID.

Delete Material Detail Line

The Delete Material Detail Line API deletes the material line in the batch.

Following is the definition of Delete Material Detail Line:

Parameter	Required	In/Out	Description
p_material_detail	Y	IN	The material detail row to be deleted.

Parameter - p_material_detail (IN)

This is a row type parameter that identifies the batch step. The following table explains the required columns of the row:

Parameter	Description
material_detail_id	Unique identifier for a batch or firm planned order line. If this is not passed, then line_no, line_type, and batch_id must be passed.
line_no	Sequential line number for each line type in a batch or firm planned order.
line_type	Valid values are -1 = Ingredient, 1 = Product; the product on Line 1 is the primary product, 2 = Byproduct.
batch_id	Batch Identifier FK to the GME_BATCH_HEADER table.

End Completed Resource Transaction

The End Completed Resource Transaction API sets the end date and resource usage for a transaction that was created using the Start Completed Resource Transaction API. This API cannot be used on a transaction that was not created with the Start Completed Resource Transaction API. This can only be done for a step that is in a work in process or completed state. The End Completed Resource Transaction API passes in the resource transaction ID, where a transaction is already started, and the end_date. The API can end only those transactions that have a value of 0 for the transaction usage, and end_date and start_date is same. This is done so that transactions cannot be ended more than once. This API cannot be run against a WIP step for a batch that has the Calculate Step Quantity indicator checked. You cannot use this API on a resource transaction that was automatically generated by the Automatic Step Quantity Calculation functionality.

Following is the definition of End Completed Resource Transaction:

Parameter	**Required**	**In/Out**	**Description**
p_poc_trans_id	Y	IN	To uniquely identify resource transactions to be ended.
p_end_date	Y	IN	End date of resource transactions.
p_reason_code	Y	IN	Reason code to create a completed transaction.
p_instance_id	N	IN	Instance ID for the transactions.
p_instance_no	N	IN	Instance number for the transactions.
p_trans_date	N	IN	The date on which the transaction is booked for financial purposes.

Delete Step

The Delete Step API deletes a step in the batch.

Following is the definition of Delete Step:

Parameter	Required	In/Out	Description
p_batch_step	Y	IN	The batch step row to identify the step.
x_batch_step	Y	OUT	The batch step that is returned, with all the data.

Parameter - p_batch_step (IN)

This is a row type parameter that identifies the batch step. The following table explains the required columns of the row:

Parameter	Description
batchstep_id	Batch step Identifier. Batch routing step number, defaults from fm_ rout_dtl. If this is not passed, then batch_id and batchstep_no are required.
batch_id	Unique identifier for batches to FPO or batch.
batchstep_no	Batch step identifier. Batch routing step number, defaults from fm_ rout_dtl.

Incremental Backflushing

The Incremental Backflushing API is a business object that lets you incrementally record production yield as it occurs and have ingredient consumption backflushed. It can only be done on WIP or completed batches off of items with a yield or consumption type of manual or incremental. You can not drive incremental backflushing off of an item that has yield or consumption type of Automatic or Automatic by Step. If any of the ingredients, products, or byproducts are not allocated, then the incremental backflushing routine returns back the unallocated material lines in the unallocated materials structure.

If batch_id is provided for the batch_header row and the material_details row, then the batch_id provided in material_details is used. Subsequently, if the material_detail_id is provided for the material_details row, then all batch_ids are ignored, and the batch_id is calculated from the specified material_detail_id.

Following is the definition of Incremental Backflushing:

Parameter	Required	In/Out	Description
p_material_details	Y	IN	The material detail row to identify the material line.
p_qty	Y	IN	The quantity to apply incrementally.
p_trans_date	Y	IN	Transaction date for the incremental backflushing transactions
p_qty_type	Y	IN	Valid values are 0 - by increment qty, 1 - new actual qty, and 2 - percentage of plan.
p_backflush_phantoms	Y	IN	Backflush the quantities to the phantoms associated.
p_ignore_shortages	Y	IN	If set to TRUE, it does not return x_ unallocated_material. This only takes effect if the GMI:Allow Negative Inventory profile option is set to 2, Warning.
p_batch_header	Y	IN	The batch header row to identify the batch.
x_batch_header	Y	OUT	The batch header that is returned, with all the data.
p_adjust_cmplt	Y	IN	Adjust completed batches.
x_unallocated_material	Y	OUT	Table of materials, if inventory shortage exists, or incomplete manual transactions exist, or unallocated items exist.

Parameter - p_batch_header (IN)

This is a row type parameter that identifies the batch header. The following table explains the required columns of the row:

Parameter	Description
batch_id	Unique identifier for batches.
batch_no	Batch number for the batch. Batch_no, plant_code, and batch_type are required if batch_id is not set.
plant_code	Organization for which the batch was created. Batch_no, plant_code, and batch_type are required if batch_id is not set.
batch_type	Type of batch. Values include 0=batch, 10=firm planned order. Batch_ no, plant_code, and batch_type are required if batch_id is not set. Incremental backflushing can not be used with firm planned orders.

Parameter - x_batch_header (OUT)

This is a row type parameter that returns the populated batch header for the created batch.

Parameter - x_unallocated_material (OUT)

This is a table type parameter that holds the information of the unallocated material, inventory shortage, or incomplete manual allocations. The following table explains how these IDs are returned:

Parameter	Description
batch_id	Unique identifier for batches.
batch_no	Batch or firm planned order number.
material_detail_id	Unique identifier for a batch or firm planned order line.
line_type	Valid values are -1 = Ingredient, 1 = Product; The product on Line 1 is the primary product, 2 = Byproduct
line_no	Sequential line number for each line type in a batch or firm planned order.
item_id	FK to the item that is a product, ingredient, or byproduct.
item_no	Item Number.
alloc_qty	Quantity that is allocated.
unalloc_qty	Quantity to be allocated.

Parameter	Description
alloc_uom	Unit of measure for ALLOC_QTY and UNALLOC_QTY also the material lines ITEM_UM.

Insert Batchstep Resource

A resource can be added for an activity of a step. A resource can be added for a step in pending, WIP, and completed step status only. If ASQC is on, then the resource cannot be added in WIP status since the process quantity cannot be inserted. Based on the step status and ASQC, all the input data is validated.

Following is the definition of Insert Batchstep Resource:

Parameter	Required	In/Out	Description
p_batch_no	N	IN	Batch number.
p_plant_code	N	IN	Organization code.
p_batchstep_no	N	IN	Step number.
p_activity	N	IN	Activity name.
p_ignore_qty_below_cap	Y	IN	This is set to True to ignore if the process quantity defined for the resource is less than the minimum capacity of the resource.
p_batchstep_resource_rec	Y	IN	Row type parameter that identifies the batchstep resource row information.
x_batchstep_resource_rec	Y	OUT	Row type parameter that returns the batchstep resource row information.

Parameter - p_batchstep_resource_rec (IN)

This is a row type parameter that identifies the batchstep resource row information. The following table explains the required columns of the row:

Parameter	Description
BATCHSTEP_ RESOURCE_ID	For update and delete APIs, this field uniquely identifies a resource. You can either provide this field or a combination of plant_code and batch_no parameters to identify a resource. This ID is generated for insert resource API.
BATCHSTEP_ ACTIVITY_ID	This is used in insert API if the combination to identify an activity is not provided through plant_code, batch_no, step_no, and activity. This is not required for update API.
COST_ANALYSIS_CODE	Cost analysis code for resource.
COST_CMPNTCLS_ID	Cost component class ID.
PRIM_RSRC_IND	Primary resource indicator.

Parameter	Description
SCALE_TYPE	Scale type.
PLAN_RSRC_COUNT	Plan resource count is used only for pending or work in process batches.
ACTUAL_RSRC_COUNT	Actual resource count is used only for work in process or completed step.
RESOURCE_QTY_UOM	Resource quantity unit of measure.
CAPACITY_UOM	Minimum or maximum capacity unit of measure.
PLAN_RSRC_USAGE	Used only for pending or work in process step.
ACTUAL_RSRC_USAGE	Used only for work in process or completed step.
PLAN_RSRC_QTY	Used only for pending or work in process step.
ACTUAL_RSRC_QTY	Used only for work in process or completed step.
USAGE_UOM	Unit of measure for resource usage fields.
PLAN_START_DATE	Required only for pending step.
ACTUAL_START_DATE	Required for work in process step.
PLAN_CMPLT_DATE	Required for pending step.
ACTUAL_CMPLT_DATE	Required for complete step.

Parameter - x_batchstep_resource_rec (OUT)

This is a row type parameter that returns the batchstep resource row information. All fields in the row are appropriately populated in the API and returned to the caller for plan and actual resource quantity. If ASQC is ON, then you cannot do an update. The following table explains the required columns of the row:

Parameter	Description
BATCHSTEP_ RESOURCE_ID	For update and delete APIs, this field uniquely identifies a resource. You can either provide this field or a combination of plant_code and batch_no parameters to identify a resource. This ID is generated for insert resource API.
BATCHSTEP_ ACTIVITY_ID	This is used in insert API if the combination to identify an activity is not provided through plant_code, batch_no, step_no, and activity. This is not required for update API.
COST_ANALYSIS_CODE	Cost analysis code for resource.

Parameter	Description
COST_CMPNTCLS_ID	Cost component class ID.
PRIM_RSRC_IND	Primary resource indicator.
SCALE_TYPE	Scale type.
PLAN_RSRC_COUNT	Plan resource count is used only for pending or work in process batches.
ACTUAL_RSRC_COUNT	Actual resource count is used only for work in process or completed step.
RESOURCE_QTY_UOM	Resource quantity unit of measure.
CAPACITY_UOM	Minimum or maximum capacity unit of measure.
PLAN_RSRC_USAGE	Used only for pending or work in process step.
ACTUAL_RSRC_USAGE	Used only for work in process or completed step.
PLAN_RSRC_QTY	Used only for pending or work in process step.
ACTUAL_RSRC_QTY	Used only for work in process or completed step.
USAGE_UOM	Unit of measure for resource usage fields.
PLAN_START_DATE	Required only for pending step.
ACTUAL_START_DATE	Required for work in process step.
PLAN_CMPLT_DATE	Required for pending step.
ACTUAL_CMPLT_DATE	Required for complete step.

Insert Incremental Completed Transaction

An actual resource transaction can be posted for a particular resource of an activity of a step. This can only be done for a step that is in a work in process or completed state. The Insert Incremental Completed Transaction API passes in trans_date, start_date, end_date, and resource usage along with the specification for the resource where a transaction needs to be added. This API adds the transaction, in addition to all previous resource transactions present for the resource. This API cannot be run against a WIP step for a batch that has the Calculate Step Quantity indicator checked.

Following is the definition of Insert Incremental Completed Transaction:

Parameter	Required	In/Out	Description
p_batchstep_rsrc_id	N	IN	To uniquely identify resource row for the transactions to be posted.
p_plant_code	N	IN	Plant where the batch exists. If you do not use p_batchstep_rsrc_id, then you must use p_plant_code, p_batch_no, p_batchstep_no, p_activity, and p_resource.
p_batch_no	N	IN	Batch for the resource transactions to be written.
p_batchstep_no	N	IN	Step number for the specific activity.
p_activity	N	IN	Activity for the specified resource.
p_resource	N	IN	Resource for the transactions to be posted.
p_trans_date	Y	IN	Date the transaction was posted on.
p_start_date	Y	IN	Start date of actual transaction. Must be within resource dates.
p_end_date	Y	IN	End date of actual transaction. Must be within resource dates and greater than p_ start_date.
p_usage	Y	IN	Resource usage can be equal to end_date. Start_date is converted in HRS.
p_reason_code	Y	IN	Reason code to create a completed transaction.
p_instance_id	N	IN	Instance ID for the transactions.
p_instance_no	N	IN	Instance number for the transactions.

Insert Line Allocation

Material lines in a batch can be set up for automatic allocation (ingredients) or manual (all types) and these allocations can be edited. The status of the batch or step, together with the release type of a line determines which quantities, planned or actual, are updated when the allocations are changed.

The Insert Line Allocation API lets you create pending or complete allocations for a given detail line in a given batch. The batch can be in a pending, WIP, or completed state and the material detail line can have any release type.

Following is the definition of Insert Line Allocation:

Parameter	Required	In/Out	Description
p_tran_row	Y	IN	The transaction row for creating the allocation.
p_lot_no	N	IN	If the lot id is not passed, then the system would use the p_lot_no to generate the lot id.
p_sublot_no	N	IN	If the lot id is not passed, then the system would use the p_lot_no, p_sublot_no to generate the lot id.
p_create_lot	N	IN	If the lot has to be created on the fly. Only valid for products, byproducts and phantom ingredients. Default is FALSE
p_ignore_shortage	N	IN	If any shortages should be ignored. This value will only be used if the GMI: Allow negative inventory profile is set to 2. Default value is FALSE.
p_scale_phantom	N	IN	If any changes to the material line quantities should be backflushed to the phantom batch. This parameter is only valid for phantom ingredients. Default value is FALSE.
x_material_detail	Y	OUT	This is the updated material line row. The possible updates could be the plan_qty, wip_plan_qty and alloc_ind depending on the batch status and the allocation made.
x_tran_row	Y	OUT	This is the updated transaction row.
x_def_tran_row	Y	OUT	This is the default transaction row for the material line, with the changes to the quantities based on the new allocation.

Parameter - p_tran_row (IN)

This is a row type parameter that identifies the transaction row information. The following table explains the required columns of the row:

Parameter	Description
batch_id	The unique identifier for the batch where the allocation is being added.
material_detail_id	The unique identifier for the material line where the allocation is being added.
lot_id	The lot where the allocation is being added. Required for lot controlled items. If the Lot ID is not passed, then the API uses the p_lot_no and p_sublot_no to derive the lot_id.
whse_code	The warehouse code where the allocation is being added.
location	The location in the warehouse where the allocation is being added. Required if the item and warehouse are location controlled.
alloc_qty	The quantity allocated in the material lines UOM. If the allocated quantity is not passed in, then the API uses the trans_qty to derive the alloc_qty. If the trans quantity is NULL, the item is a dual UOM controlled 1 or 2 type of item, and the trans_qty2 is passed in, then the API uses the trans_qty2 to calculate the alloc_qty. Note the quantity passed in must be positive.
trans_qty	The quantity allocated in the items primary UOM. This quantity is ignored if the alloc_qty is passed in.
trans_qty2	The quantity allocated in the items secondary UOM. If it is not passed in, then the alloc_qty is used to derive it for dual UOM 1 or 2 type of items.
completed_ind	Valid values are 0 - for creating pending allocation, 1 - for creating completed allocation. This field is only used if the material release type is manual or incremental.
trans_date	The transaction date for the allocation. If the trans_date is not passed, then the system uses the default rules based on the batch status or step status, line type, and release type to determine the trans_date.
reason_code	The reason code associated with the allocation.

Parameter - x_material_detail (OUT)

This is a row type parameter that returns the updated material line information.

Parameter - x_tran_row (OUT)

This is a row type parameter that returns the updated transaction row information.

Parameter - x_def_tran_row (OUT)

This is a row type parameter that returns the default transaction row information with any adjustments.

Insert Material Detail Line

The Insert Material Detail Line API is used to insert ingredients, products, or byproducts into a batch.

Following is the definition of Insert Material Detail Line:

Parameter	Required	In/Out	Description
p_material_detail	Y	IN	The material detail row to insert the material line.
p_batchstep_no	N	IN	The batch step that the material line is associated to, if any. Default value is NULL.
x_material_detail	Y	OUT	Inserted material line.

Parameter - p_material_detail (IN)

This is a row type parameter that identifies the material detail. The following table explains the required columns of the row:

Parameter	Description
batch_id	Batch identifier foreign key to the GME_BATCH_HEADER table.
line_no	Sequential line number for each line type in a batch or firm planned order.
item_id	FK to the item that is a product, ingredient, or byproduct.
line_type	Valid values are -1 = Ingredient, 1 = Product; the product on Line 1 is the primary product, 2 = Byproduct
plan_qty	The quantity that this line was planned to produce or consume, in the batch UOM (item_um column). This is only valid if the batch is in pending status, otherwise the value must be NULL.
item_um	UOM in which planned and actual quantity is entered on the batch.
actual_qty	Sum (trans_qty) of completed transactions for line in the batch UOM. This is only valid for lab batches without inventory.
release_type	Release (Consumption for ingredients and yield for product) type. Profile option determines the default value, the profile defaults to Automatic. Valid values are 0 = Automatic release (completion aka certification), 1 = Manual release (completion aka certification), 2 = Incremental release, 3 = Automatic by Step.

Parameter	Description
scrap_factor	Scrap factor decimal used to a trans_qty that allows for scrap. This is only passed in for ingredient lines.
scale_type	Valid values are 0 = Fixed, 1 = Proportional, 2 = Scale by increment.
phantom_type	Phantom indicator. Valid values are 0 = not a phantom, 1 = automatic phantom replacement, 2 = manual phantom.
cost_alloc	For products, fraction of cost allocated to this product.
text_code	ID which joins any rows of text in the table to the text table for this application.
rounding_direction	Determines whether to round up or round down to the nearest SCALE_MULTIPLE. Valid values are 0 = UP, 1 = DOWN, 2 = EITHER. Only meaningful when scale_type equals 2.
scale_rounding_variance	Percentage plus or minus of the scaled quantity variance allowed when scaling; default is zero. Only meaningful when scale_type equals 2.
scale_multiple	The multiples of the batch uom (scale_uom) for scaling. Only meaningful when scale_type equals 2.
contribute_yield_ind	Indicates if the ingredients contribute to yield. If the item is defined as a packaging item, then the item cannot contribute to yield. Therefore, the value is N, No. Valid values are Y = The item contributes to yield, Default, N = No, the item does not contributes to yield.
contribute_step_qty_ind	Indicates if the ingredients contribute to step quantity. If item is defined as a packaging item, then the item cannot contribute to the step quantity. Therefore, the value is N, No. Valid values are Y= The item contributes to step quantity, Default value, N = No, the item does not contributes to step quantity.
wip_plan_qty	This quantity is used in place of PLAN_QTY, if the batch status is WIP or above. This is required if the batch is in WIP status.

Parameter - x_material_detail (OUT)

This is a row type parameter that returns the material detail for the inserted line.

Insert Step

The Insert Step API inserts a new step into a batch. To insert a step, you must pass in an operation.

Following is the definition of Insert Step:

Parameter	Required	In/Out	Description
p_batch_step	Y	IN	The batch step row to identify the step.
x_batch_step	Y	OUT	The batch step that is returned, with all the data.
p_batch_header	Y	IN	The batch header associated to the batch step.

Parameter - p_batch_step (IN)

This is a row type parameter that identifies the batch step. The following table explains the required columns of the row:

Parameter	Description
batch_id	Unique identifier for batches to FPO or batch.
batchstep_no	Batch step identifier. Batch routing step number, defaults from fm_ rout_dtl.
oprn_id	Foreign key to the operation GMD_OPERATIONS, defaults from fm_ rout_dtl. Stored for all batch steps. If a routing step is added to batch not originally defined to the recipe, the oprn_id is stored. Stored for steps not defined in routing.
plan_step_qty	If scaling and auto-calculate step quantity equals On and formula is scalable, then first you scale materials, then recalculate step quantity using step material association. If this exists, then GMD_ calculation function is used. If step dependencies exist, then consider what has flowed through from previous step. If auto-calculate step quantity is Off, then scale routing step quantities. Note that if the planned quantity in batch is changed, then the plan_step_qty is recalculated.
actual_step_qty	The actual is not stored when the step is created. Either a Null or zero is stored; will store the value stored today.
plan_start_date	Derived from GMD_OPERATIONS; stored here since the operation uom can change after the routing is created and associated with a recipe and batch.

Parameter	Description
actual_start_date	Updated at the time the step is released; defaults to system date but can be overridden. This validates the actual_start_date is batch start date; and after any step it is dependent on, provides a warning.
due_date	Required step completion date and time.
plan_cmplt_date	Based on the step planned start date and plus the time required for step to complete.
actual_cmplt_date	Updated at the time the system is completed. Defaults to system date and time and can be overridden.
step_close_date	Updated at the time the step is closed; Defaults to system date and time and can be overridden.
step_status	Valid values are 1 = Pending, 2 = WIP, 3 = Completed (Certified), 4 = Closed, 5 = Cancelled.
steprelease_type	Defaults from routing. Valid values are 1 - Manual, 2 - Automatic.
text_code	The ID that joins any rows of text in this table to the Text Table for this application.

Parameter - x_batch_step (OUT)

This is a row type parameter that returns the populated batch step for the inserted step.

Insert Timed Resource Transaction

An actual resource transaction can be posted for a particular resource of an activity of a step. This can only be done for a step that is in a work in process or completed state. The Insert Timed Resource Transaction API passes in trans_date, start_date, and end_date, along with the specification for the resource where a transaction needs to be added. This API calculates the resource usage based on the difference of p_end_date and p_start_date converted in the resource unit of measure.

Following is the definition of Insert Timed Resource Transaction:

Parameter	Required	In/Out	Description
p_batchstep_rsrc_id	N	IN	To uniquely identify resource row for which transactions need to be posted.
p_plant_code	N	IN	Plant where the batch exists. If you do not use p_batchstep_rsrc_id, then you must use p_plant_code, p_batch_no, p_batchstep_no, p_activity, and p_resource.
p_batch_no	N	IN	Batch for the resource transactions to be written.
p_batchstep_no	N	IN	Step number for the specific activity.
p_activity	N	IN	Activity for the specified resource.
p_resource	N	IN	Resource for the transactions to be posted.
p_trans_date	Y	IN	Date the transaction was posted on.
p_start_date	Y	IN	Start date of actual transaction. Must be within resource dates.
p_end_date	Y	IN	End date of actual transaction. Must be within resource dates and greater than p_ start_date.
p_reason_code	Y	IN	Reason code to create a completed transaction.
p_instance_id	N	IN	Instance ID for the transactions.
p_instance_no	N	IN	Instance number for the transactions.

Release Batch

The Release Batch API is a business object that converts pending batches to work in process (WIP) batches in OPM.

Batch release is affected by calling the API with a set of parameters that indicate which batch to release and also an indication of whether the release can proceed in the presence of inventory shortages.

The API applies equally to batches and lab batches, but any attempt to release a firm planned order is rejected, as are attempts to release batches or lab batches that have a status other than pending.

If the batch or lab batch is released successfully, then the API returns with a status of S. If errors occur, then the return status is E for errors that are normal but prevent the batch from being released, and U for errors that are unexpected and also prevent the batch from being released, such as database errors.

On successful release, an updated batch header is returned to the caller with various values updated (batch_status, actual_start_date) so there is immediate knowledge of the new data.

Following is the definition of Release Batch:

Parameter	Required	In/Out	Description
p_ignore_shortages	Y	IN	If set to TRUE, it does not return x_ unallocated_material. This only takes effect if the GMI:Allow Negative Inventory profile options is set to 2, Warning.
p_batch_header	Y	IN	The batch header row to identify the batch.
x_batch_header	Y	OUT	The batch header that is returned, with all the data.
x_unallocated_material	Y	OUT	Table of materials, if inventory shortage exists, or unallocated items exist.
p_ignore_unalloc	Y	IN	Proceed with the release even if there are unallocated ingredients with a consumption type of automatic. Unallocated quantities will not be consumed.

Parameter - p_batch_header (IN)

This is a row type parameter that identifies the batch header. The following table explains the required columns of the row:

Parameter	Description
batch_id	Unique identifier for batches.
batch_no	Batch number for the batch. Batch_no, plant_code, and batch_type are required if batch_id is not set.
plant_code	Organization for which the batch was created. Batch_no, plant_code, and batch_type are required if batch_id is not set.
batch_type	Type of batch. Values include 0=batch, 10=firm planned order. Batch_no, plant_code, and batch_type are required if batch_id is not set.
actual_start_date	Actual start date and time for batch. If no date is passed in, it defaults to the current system date and time.

Parameter - x_batch_header (OUT)

This is a row type parameter that returns the populated batch header for the released batch.

Parameter - x_unallocated_material (OUT)

This is a table type parameter that holds the information of the unallocated material, inventory shortage, or incomplete manual allocations. The following table explains how these IDs are returned:

Parameter	Description
batch_id	Unique identifier for batches.
batch_no	Batch or firm planned order number.
material_detail_id	Unique identifier for a batch or firm planned order line.
line_type	Valid values are -1 = Ingredient, 1 = Product, 2 = Byproduct
line_no	Sequential line number for each line type in a batch or firm planned order.
item_id	FK to the item that is a product, ingredient, or byproduct.
item_no	Item Number.
alloc_qty	Quantity that is allocated.
unalloc_qty	Quantity that was unallocated.
alloc_uom	Unit of measure for ALLOC_QTY and UNALLOC_QTY also the material lines ITEM_UM.

Release Steps

The Release Step API is a business object that can release a single batch step or multiple batch steps based on certain criteria. Releasing a step is how input quantities are specified, or defaulted, for items used in the step. The API operates on steps that have a status of pending and if prior steps are release type of automatic, then these steps in the dependency chain are released before the selected step is released. If the batch is pending and the GME:Step Controls Batch Status profile option is set to Yes, then the batch is released prior to releasing the step.

If an ingredient is associated to the step and has a consumption type of Automatic by Step, then that ingredient line is released. If that ingredient is not allocated, then there is an attempt to allocate. Unallocated items and inventory shortages are reported at this time.

Only batch steps in pending can be released. The batch step that is passed in must contain sufficient information to identify the batch step (either batch_id and batchstep_no or batchstep_id).

Following is the definition of Release Step:

Parameter	Required	In/Out	Description
p_ignore_shortages	N	IN	If set to TRUE, it does not return x_ unallocated_material. This only takes effect if the GMI:Allow Negative Inventory profile options is set to 2, Warning.
p_batch_step	Y	IN	The batch step row to identify the batch.
x_batch_step	Y	OUT	The batch step that is returned, with all the data.
x_unallocated_material	Y	OUT	Table of materials, if inventory shortage exists, or unallocated items exist.
p_ignore_unalloc	Y	IN	Do not check for the item requiring allocations.

Parameter - p_batch_step (IN)

This is a row type parameter that identifies the batch step. The following table explains the required columns of the row:

Parameter	Description
batchstep_id	Unique identifier for batch steps.

Parameter	Description
batch_id	ID of the batch. Batch_id and batchstep_no are required if batchstep_id is not set.
batchstep_no	Step number. Batch_id and batchstep_no are required if batchstep_id is not set.
actual_start_date	Actual start date and time for batch step. If no date is passed in, it defaults to the current system date and time.

Parameter - x_batch_step (OUT)

This is a row type parameter that returns the populated batch step for the completed step.

Parameter - x_unallocated_material (OUT)

This is a table type parameter that holds the information of the unallocated material, inventory shortage, or incomplete manual allocations. The following table explains how these IDs are returned:

Parameter	Description
batch_id	Unique identifier for batches to firm planned order or batch.
batch_no	Batch or firm planned order number.
material_detail_id	Unique identifier for a batch or firm planned order line.
line_type	Valid values are -1 = Ingredient, 1 = Product; The product on Line 1 is the primary product, 2 = Byproduct
line_no	Sequential line number for each line type in a batch or firm planned order.
item_id	FK to the item that is a product, ingredient, or byproduct.
item_no	Item Number.
alloc_qty	Quantity that is allocated.
unalloc_qty	Quantity to be allocated.
alloc_uom	Unit of measure for ALLOC_QTY and UNALLOC_QTY also the material lines ITEM_UM.

Reopen Batch

Reopening a batch changes the status of the batch from closed to complete.

Only batches in a closed state can be reopened. The batch header that is passed in must contain sufficient information to identify the batch (firm planned orders are not eligible for reopening). You cannot reopen a batch where the transactions have been purged, the batch has been migrated, the actual cost has been run and frozen, or the GL posting is complete.

Following is the definition of Reopen Batch:

Parameter	Required	In/Out	Description
p_batch_header	Y	IN	The batch header row to identify the batch.
x_batch_header	Y	OUT	The batch header that is returned, with all the data.
p_reopen_steps	N	IN	Reopen all the steps. Default value is False.

Parameter - p_batch_header (IN)

This is a row type parameter that identifies the batch header. The following table explains the required columns of the row:

Parameter	Description
batch_id	Unique identifier for batches to firm planned order or batch.
batch_no	Batch number for the batch. Batch_no, plant_code, and batch_type are required if batch_id is not set.
plant_code	Organization for which the batch was created. Batch_no, plant_code, and batch_type are required if batch_id is not set.
batch_type	Type of batch. Values include 0=batch, 10=firm planned order. Batch_no, plant_code, and batch_type are required if batch_id is not set.

Parameter - x_batch_header (OUT)

This is a row type parameter that returns the populated batch header for the reopened batch.

Reopen Steps

Reopening a batch step changes the status of the step from close to completed and updates the step_close_date to Null.

This API only reopens the specified batch step (passed parameter).

Reopening a batch step can be done only on a batch that is not closed. The batch step that is passed in must contain sufficient information to identify at least batch_step_id.

Following is the definition of Reopen Step:

Parameter	Required	In/Out	Description
p_batch_step	Y	IN	The batch step row to identify the batch step.
x_batch_step	Y	OUT	The batch step that is returned, with all the data.

Parameter - p_batch_step (IN)

This is a row type parameter that identifies the batch step. The following table explains the required columns of the row:

Parameter	Description
batchstep_id	Unique identifier for batch steps.
batch_id	ID of the batch. Batch_id and batchstep_no are required if batchstep_id is not set.
batchstep_no	Step number. Batch_id and batchstep_no are required if batchstep_id is not set.

Parameter - x_batch_step (OUT)

This is a row type parameter that returns the populated batch step for the reopened step.

Reroute Batch

The Reroute Batch API is a business object that can reroute a batch or firm planned order to a different recipe with the same formula. Batches can only be rerouted in pending status.

Following is the definition of Reroute Batch:

Parameter	Required	In/Out	Description
p_batch_header	Y	IN	The batch header row to identify the batch.
x_batch_header	Y	OUT	The batch header that is returned, with all the data.
p_validity_rule_id	Y	IN	Recipe validity rule ID for the new recipe.

Parameter - p_batch_header (IN)

This is a row type parameter that identifies the batch header. The following table explains the required columns of the row:

Parameter	Description
batch_id	Unique identifier for batches to firm planned order or batch.
batch_no	Batch number for the batch. Batch_no, plant_code, and batch_type are required if batch_id is not set.
plant_code	Organization for which the batch was created. Batch_no, plant_code, and batch_type are required if batch_id is not set.
batch_type	Type of batch. Values include 0=batch, 10=firm planned order. Batch_no, plant_code, and batch_type are required if batch_id is not set.

Parameter - x_batch_header (OUT)

This is a row type parameter that returns the populated batch header for the rerouted batch.

Reschedule Batch

The Reschedule Batch API is a business object that can reschedule a batch or firm planned order to different date. A batch can only be rescheduled while in pending or WIP state. A firm planned order can only be rescheduled when in a pending state.

Following is the definition of Reschedule Batch:

Parameter	Required	In/Out	Description
p_batch_header	Y	IN	The batch header row to identify the batch.
x_batch_header	Y	OUT	The batch header that is returned, with all the data.

Parameter - p_batch_header (IN)

This is a row type parameter that identifies the batch header. The following table explains the required columns of the row:

Parameter	Description
batch_id	Unique identifier for batches to firm planned order or batch.
batch_no	Batch number for the batch. Batch_no, plant_code, and batch_type are required if batch_id is not set.
plant_code	Organization for which the batch was created. Batch_no, plant_code, and batch_type are required if batch_id is not set.
batch_type	Type of batch. Values include 0=batch, 10=firm planned order. Batch_ no, plant_code, and batch_type are required if batch_id is not set.
plan_start_date	New planned start date and time for batch. This is passed in by you, and can only be specified if the batch is pending.
plan_cmplt_date	New planned completion date and time for batch. This is passed in by you.

Parameter - x_batch_header (OUT)

This is a row type parameter that returns the populated batch header for the rescheduled batch.

Reschedule Step

The Reschedule Step API reschedules the step and all subsequent steps, if requested.

Following is the definition of Reschedule Step:

Parameter	Required	In/Out	Description
p_batch_step	Y	IN	The batch step row to identify the batch.
x_batch_step	Y	OUT	The batch step that is returned, with all the data.
p_reschedule_other	N	IN	Determines whether to reschedule other steps in a step dependency chain.

Parameter - p_batch_step (IN)

This is a row type parameter that identifies the batch step. The following table explains the required columns of the row:

Parameter	Description
batch_id	ID of the batch. Batch_id and batchstep_no are required if batchstep_ id is not set.
batchstep_no	Step number. Batch_id and batchstep_no are required if batchstep_id is not set.
batchstep_id	Unique identifier for batch steps.
plan_start_date	New planned start date and time for step. This is passed in by you, and can only be specified if the step is pending.
plan_cmplt_date	New planned completion date and time for step. This is passed in by you.

Parameter - x_batch_step (OUT)

This is a row type parameter that returns the populated batch step for the rescheduled step.

Revert to WIP Batch

The Revert to WIP Batch, or uncertify batch, API is a business object that can uncomplete the batch. Reverting a batch to WIP changes transactions from completed back to pending for products and byproducts with a yield type of Automatic, and changes the batch status back to WIP.

Only batches in completed state can be reverted to QIP. The batch header that is passed in must contain sufficient information to identify the batch. Firm planned orders are not eligible for completing or reverting to WIP.

Following is the definition of Uncertify Batch:

Parameter	Required	In/Out	Description
p_batch_header	Y	IN	The batch header row to identify the batch.
x_batch_header	Y	OUT	The batch header that is returned, with all the data.

Parameter - p_batch_header (IN)

This is a row type parameter that identifies the batch header. The following table explains the required columns of the row:

Parameter	Description
batch_id	Unique identifier for batches.
batch_no	Batch number for the batch. Batch_no, plant_code, and batch_type are required if batch_id is not set.
plant_code	Organization for which the batch was created. Batch_no, plant_code, and batch_type are required if batch_id is not set.
batch_type	Type of batch. Values include 0=batch, 10=firm planned order. Batch_no, plant_code, and batch_type are required if batch_id is not set.

Parameter - x_batch_header (OUT)

This is a row type parameter that returns the populated batch header for the created batch.

Revert to WIP Steps

The Revert to WIP Steps, or uncertify batch steps, API is a business object that can uncomplete a batch step based on certain criteria. Reverting a step to WIP puts the step back to WIP status and posts pending transactions after posting reversing completed transactions for the product lines associated with the step and with a consumption type of Automatic by Step.

To revert a step to WIP, any steps that are dependent on this step and have a step release type of Automatic are pending or WIP.

Only batch steps in the status of complete can be reverted to WIP. Also, the batch can be in pending or WIP status.

Following is the definition of Uncertify Step:

Parameter	Required	In/Out	Description
p_batch_step	Y	IN	The batch step row to identify the batch step.
x_batch_step	Y	OUT	The batch step that is returned, with all the data.

Parameter - p_batch_step (IN)

This is a row type parameter that identifies the batch step. The following table explains the required columns of the row:

Parameter	Description
batch_id	ID of the batch. Batch_id and batchstep_no are required if batchstep_id is not set.
batchstep_no	Step number. Batch_id and batchstep_no are required if batchstep_id is not set.
batchstep_id	Unique identifier for batch steps.

Parameter - x_batch_step (OUT)

This is a row type parameter that returns the populated batch step for the completed step.

Scale Batch

The Scale Batch API scales batches up or down, including phantom batches.

Following is the definition of Scale Batch:

Parameter	Required	In/Out	Description
p_scale_factor	Y	IN	How much to scale. Scale multiplier to make twice as much quantity, scale factor = 2, to reduce quantity to half scale factor = -0.5.
p_primaries	Y	IN	Scaling based on products or ingredients. Valid values are INPUT = ingredients, OUTPUT = products.
p_batch_header	Y	IN	The batch header row to identify the batch.
x_batch_header	Y	OUT	The batch header that is returned, with all the data.
x_over_allocations	Y	OUT	Tables of material lines, trying to scale down the batch, and the quantities are going below allocations.

Parameter - p_batch_header (IN)

This is a row type parameter that identifies the batch header. The following table explains the required columns of the row:

Parameter	Description
batch_id	Unique identifier for batches to firm planned order or batch.
batch_no	Batch number for the batch. Batch_no, plant_code, and batch_type are required if batch_id is not set.
plant_code	Organization for which the batch was created. Batch_no, plant_code, and batch_type are required if batch_id is not set.
batch_type	Type of batch. Values include 0=batch, 10=firm planned order. Batch_no, plant_code, and batch_type are required if batch_id is not set.

Parameter - x_batch_header (OUT)

This is a row type parameter that returns the populated batch header for the scaled batch.

Parameter - x_unallocated_material (OUT)

This is a table type parameter that holds the information of the unallocated material, inventory shortage, or incomplete manual allocations. The following table explains how these IDs are returned:

Parameter	Description
batch_id	Unique identifier for batches.
batch_no	Batch or firm planned order number.
material_detail_id	Unique identifier for a batch or firm planned order line.
line_type	Valid values are -1 = Ingredient, 1 = Product; The product on Line 1 is the primary product, 2 = Byproduct
line_no	Sequential line number for each line type in a batch or firm planned order.
item_id	FK to the item that is a product, ingredient, or byproduct.
item_no	Item Number.
alloc_qty	Quantity that is allocated.
unalloc_qty	Quantity that was unallocated.
alloc_uom	Unit of measure for ALLOC_QTY and UNALLOC_QTY also the material lines ITEM_UM.

Start Completed Resource Transaction

The Start Completed Resource Transaction API is used in conjunction with the End Completed Resource Transaction API. An actual resource transaction can be posted for a particular resource of an activity of a step with the start date of the transaction. This can only be done for a step that is in a work in process or completed state. The Start Completed Resource Transaction API passes in trans_date, and start_date, along with the specification for the resource where the transaction needs to be added. This API puts the start_date value as the end_date. Therefore, usage for the resource transaction is 0. When the End Completed Resource Transaction API is run, the usage is calculated based on the difference between the start date and the end date. This API cannot be run against a WIP step for a batch that has the Calculate Step Quantity indicator checked. It also passes the poc_trans_id that gets generated. This ID can be for the End Completed Resource Transaction API for ending the same transaction.

Following is the definition of Start Completed Resource Transaction:

Parameter	Required	In/Out	Description
p_batchstep_rsrc_id	N	IN	To uniquely identify resource row for the transaction to be posted.
p_plant_code	N	IN	Plant in which the batch exists. If you do not use p_batchstep_rsrc_id, then you must use p_plant_code, p_batch_no, p_batchstep_no, p_activity, and p_resource.
p_batch_no	N	IN	Batch for the resource transaction to be written.
p_batchstep_no	N	IN	Step number for the activity to be specified.
p_activity	N	IN	Activity for the resource to be specified.
p_trans_date	Y	IN	Date the transaction was posted on.
p_start_date	Y	IN	Start date of actual transaction. Must be within resource dates. Same date is put in end_date.
p_reason_code	Y	IN	Reason code to create a completed transaction.
p_instance_id	N	IN	Instance ID for the transaction.
p_instance_no	N	IN	Instance number for the transaction.
x_rsrc_txn_id	N	OUT	To uniquely identify resource transaction to be posted with an end date.

Theoretical Yield Batch

The Theoretical Yield Batch API calculates theoretical yield for the batch, and updates the quantities for the product lines.

Following is the definition of Theoretical Yield Batch:

Parameter	Required	In/Out	Description
p_scale_factor	Y	IN	Theoretical yield in fractions. For example, to set a 90% yield, set the scale factor to.9.
p_batch_header	Y	IN	The batch header row to identify the batch.
x_batch_header	Y	OUT	The batch header that is returned, with all the data.

Parameter - p_batch_header (IN)

This is a row type parameter that identifies the batch header. The following table explains the required columns of the row:

Parameter	Description
batch_id	Unique identifier for batches to firm planned order or batch.
batch_no	Batch number for the batch. Batch_no, plant_code, and batch_type are required if batch_id is not set.
plant_code	Organization for which the batch was created. Batch_no, plant_code, and batch_type are required if batch_id is not set.
batch_type	Type of batch. Values include 0=batch, 10=firm planned order. Batch_no, plant_code, and batch_type are required if batch_id is not set.

Parameter - x_batch_header (OUT)

This is a row type parameter that returns the populated batch header for the theoretical yield batch.

Unrelease Batch

The Unrelease Batch API reverses the completed transactions for the ingredient lines. Unreleasing a batch sets the batch status to pending.

Only batches in WIP state can be unreleased. The batch header that is passed in must contain sufficient information to identify the batch. Firm planned orders are not eligible for unreleasing.

Following is the definition of Unrelease Batch:

Parameter	Required	In/Out	Description
p_batch_header	Y	IN	The batch header row to identify the batch.
x_batch_header	Y	OUT	The batch header that is returned, with all the data.
p_preserve_allocations	Y	IN	Preserve lot allocations.

Parameter - p_batch_header (IN)

This is a row type parameter that identifies the batch header. The following table explains the required columns of the row:

Parameter	Description
batch_id	Unique identifier for batches.
batch_no	Batch number for the batch. Batch_no, plant_code, and batch_type are required if batch_id is not set.
plant_code	Organization for which the batch was created. Batch_no, plant_code, and batch_type are required if batch_id is not set.
batch_type	Type of batch. Values include 0=batch, 10=firm planned order. Batch_no, plant_code, and batch_type are required if batch_id is not set.

Parameter - x_batch_header (OUT)

This is a row type parameter that returns the populated batch header for the released batch.

Unrelease Steps

The Unrelease Steps API is a business object that can unrelease a batch step based on certain criteria. Unreleasing a step sets the step back to pending status, and reverses completed transactions for any ingredients consumed in the step that have a consumption type of Automatic by Step.

Following is the definition of Unrelease Step:

Parameter	Required	In/Out	Description
p_batch_step	Y	IN	The batch step row to identify the batch.
x_batch_step	Y	OUT	The batch step that is returned, with all the data.
p_preserve_allocations	Y	IN	Preserve lot allocations.

Parameter - p_batch_step (IN)

This is a row type parameter that identifies the batch step. The following table explains the required columns of the row:

Parameter	Description
batch_id	ID of the batch. Batch_id and batchstep_no are required if batchstep_id is not set.
batchstep_no	Step number. Batch_id and batchstep_no are required if batchstep_id is not set.
batchstep_id	Unique identifier for batch steps.

Parameter - x_batch_step (OUT)

This is a row type parameter that returns the populated batch step for the released step.

Update Actual Resource Usage

An actual resource transaction can be posted for a particular resource of an activity of a step. This can only be done for a step that is in a work in process or completed state. The Update Actual Resource Usage API passes in trans_date, start_date, end_date, and resource usage, along with the specification for the resource where a transaction needs to be added. This API removes all other resource transactions present for the resource.

Following is the definition of Update Actual Resource Usage:

Parameter	Required	In/Out	Description
p_batchstep_rsrc_id	N	IN	To uniquely identify resource row for the transactions to be posted.
p_plant_code	N	IN	Plant where the batch exists. If you do not use p_batchstep_rsrc_id, then you must use p_plant_code, p_batch_no, p_batchstep_no, p_activity, and p_resource.
p_batch_no	N	IN	Batch for the resource transactions to be written.
p_batchstep_no	N	IN	Step number for the specific activity.
p_activity	N	IN	Activity for the specified resource.
p_resource	N	IN	Resource for the transactions to be posted.
p_trans_date	Y	IN	Date the transaction was posted on.
p_start_date	Y	IN	Start date of actual transaction. Must be within resource dates.
p_end_date	Y	IN	End date of actual transaction. Must be within resource dates and greater than p_ start_date.
p_usage	Y	IN	Resource usage can be equal to end_date. Start_date is converted in HRS.
p_reason_code	Y	IN	Reason code to create a completed transaction.
p_instance_id	N	IN	Instance ID for the transactions.
p_instance_no	N	IN	Instance number for the transactions.

Update Batchstep Resource

An existing resource can be updated for an activity of a step. A resource can be updated for a step in pending, WIP, and completed step status only. Based on the step status and ASQC, all the input data is validated before the resource in updated. Based on the step status, different fields of the resource can be updated.

If any column must be updated to NULL, then you must pass in FND_API.G_MISS_CHAR, FND_API.G_MISS_NUM, or FND_API.G_MISS_DATE variables to the API column values to update the column to NULL in the database.

Following is the definition of Update Batchstep Resource:

Parameter	Required	In/Out	Description
p_batch_no	N	IN	Batch number.
p_plant_code	N	IN	Organization code.
p_batchstep_no	N	IN	Step number.
p_activity	N	IN	Activity name.
p_resource	N	IN	Resource name.
p_ignore_qty_below_cap	Y	IN	This is set to True to ignore if the process quantity defined for the resource is less than the minimum capacity of the resource.
p_batchstep_resource_rec	Y	IN	Row type parameter that identifies the batchstep resource row information.
x_batchstep_resource_rec	Y	OUT	Row type parameter that returns the batchstep resource row information.

Parameter - p_batchstep_resource_rec (IN)

This is a row type parameter that identifies the batchstep resource row information. The following table explains the required columns of the row:

Parameter	Description
batchstep_resource_id	For update and delete APIs, this field uniquely identifies a resource. You can either provide this field or a combination of plant_code and batch_no parameters to identify a resource. This ID is generated for insert resource API.

Parameter	Description
batchstep_activity_id	This is used in insert API if the combination to identify an activity is not provided through plant_code, batch_no, step_no, and activity. This is not required for update API.
cost_analysis_code	Cost analysis code for resource.
cost_cmpntcls_id	Cost component class ID.
prim_rsrc_ind	Primary resource indicator.
scale_type	Scale type.
plan_rsrc_count	Plan resource count is used only for pending or work in process batches.
actual_rsrc_count	Actual resource count is used only for work in process or completed step.
resource_qty_uom	Resource quantity unit of measure.
capacity_uom	Minimum or maximum capacity unit of measure.
plan_rsrc_usage	Used only for pending or work in process step.
actual_rsrc_usage	Used only for work in process or completed step.
plan_rsrc_qty	Used only for pending or work in process step.
actual_rsrc_qty	Used only for work in process or completed step.
usage_uom	Unit of measure for resource usage fields.
plan_start_date	Required only for pending step.
actual_start_date	Required for work in process step.
plan_cmplt_date	Required for pending step.
actual_cmplt_date	Required for complete step.

Parameter - x_batchstep_resource_rec (OUT)

This is a row type parameter that returns the batchstep resource row information. All fields in the row are appropriately populated in the API and returned to the caller for plan and actual resource quantity. If ASQC is ON, then you cannot do an update. The following table explains the required columns of the row:

Parameter	Description
batchstep_resource_id	For update and delete APIs, this field uniquely identifies a resource. You can either provide this field or a combination of plant_code and batch_no parameters to identify a resource. This ID is generated for insert resource API.
batchstep_activity_id	This is used in insert API if the combination to identify an activity is not provided through plant_code, batch_no, step_no, and activity. This is not required for update API.
cost_analysis_code	Cost analysis code for resource.
cost_cmpntcls_id	Cost component class ID.
prim_rsrc_ind	Primary resource indicator.
scale_type	Scale type.
plan_rsrc_count	Plan resource count is used only for pending or work in process batches.
actual_rsrc_count	Actual resource count is used only for work in process or completed step.
resource_qty_uom	Resource quantity unit of measure.
capacity_uom	Minimum or maximum capacity unit of measure.
plan_rsrc_usage	Used only for pending or work in process step.
actual_rsrc_usage	Used only for work in process or completed step.
plan_rsrc_qty	Used only for pending or work in process step.
actual_rsrc_qty	Used only for work in process or completed step.
usage_uom	Unit of measure for resource usage fields.
plan_start_date	Required only for pending step.
actual_start_date	Required for work in process step.
plan_cmplt_date	Required for pending step.
actual_cmplt_date	Required for complete step.

Update Material Detail Line

The Update Material Detail Line API updates the material line in the batch.

Following is the definition of Update Material Detail Line:

Parameter	Required	In/Out	Description
p_material_detail	Y	IN	The material detail row to insert the material line.
x_material_detail	Y	OUT	The updated material detail row.
p_values_tab	Y	IN	The columns in the material line which can be updated.
p_scale_phantom	N	IN	Indicates if the quantity must be backflushed to the phantom batch. Default value is False.

Parameter - p_material_detail (IN)

This is a row type parameter that identifies the batch step. The following table explains the required columns of the row:

Parameter	Description
material_detail_id	Unique identifier for a batch or firm planned order line.
batch_id	Batch identifier foreign key to the GME_BATCH_HEADER table.
plan_qty	The quantity that this line was planned to produce or consume, in the batch UOM (item_um column). This is used only if the batch is pending.
actual_qty	Sum (trans_qty) of completed transactions for line in the batch UOM. This is used only for lab batches without inventory.
release_type	Release (Consumption for ingredients and yield for product) type. Profile option determines the default value, the profile defaults to Automatic. Valid values are 0 = Automatic release (completion aka certification), 1 = Manual release (completion aka certification), 2 = Incremental release, 3 = Automatic by Step. This is valid only if the batch is pending.
scrap_factor	Scrap factor decimal used to a trans_qty that allows for scrap. Updated only if the batch is pending, and valid only for ingredients.
scale_type	Valid values are 0 = Fixed, 1 = Proportional, 2 = Scale by increment.

Parameter	Description
phantom_type	Phantom indicator. Valid values are 0 = not a phantom, 1 = automatic phantom replacement, 2 = manual phantom. This is valid only if the batch is pending and the phantom is not exploded.
cost_alloc	For products, fraction of cost allocated to this product.
text_code	ID which joins any rows of text in the table to the text table for this application.
rounding_direction	Determines whether to round up or round down to the nearest SCALE_MULTIPLE. Valid values are 0 = UP, 1 = DOWN, 2 = EITHER. This is meaningful only if scale_type equals 2.
scale_rounding_variance	Percentage plus or minus of the scaled quantity variance allowed when scaling; default is zero. This is meaningful only if scale_type equals 2.
scale_multiple	The multiples of the batch uom (scale_uom) for scaling. This is meaningful only if scale_type equals 2.
contribute_yield_ind	Indicates if the item (product or ingredient) contributes to yield. If the item is defined as a packaging item, then the item cannot contribute to yield. Therefore, the value is N, No. Valid values are Y = The item contributes to yield, Default, N = No, the item does not contributes to yield.
contribute_step_qty_ind	Indicates if the item contributes to step quantity. If item is defined as a packaging item, then the item cannot contribute to the step quantity. Therefore, the value is N, No. Valid values are Y= The item contributes to step quantity, Default value, N = No, the item does not contributes to step quantity.
wip_plan_qty	This quantity is used in place of PLAN_QTY, if the batch status is WIP or above.

Parameter - x_material_detail (OUT)

This is a row type parameter that returns the populated batch step for the updated line.

A Messages and Errors

Handling Messages

APIs put result messages into a message list. Programs calling APIs can then get the messages from the list and process them by either issuing them, loading them in a database table, or writing them to a log file. Messages are stored in an encoded format to enable API callers to find out message names by using the standard functions provided by the message dictionary. It also stores these messages in database tables and reports off these tables in different languages. The structure of the message list is not public. Neither API developers nor API callers can access this list except through calling the API message utility routines mentioned below.

The following utility functions are defined in the FND_MSG_PUB package, in the file AFASMSGS.pls:

Initialize Initializes the API message list

Add Adds a message to the API message list

Get Gets a message from the API message list

Count_Msg Returns the number of messages in the API message list

Delete Deletes one or more messages from the API message list

Reset Resets the index used in getting messages

Count_And_Get Returns the number of messages in the API message list. If this number is one, then it also returns the message data.

To add a message to the API message list, developers can use the regular message dictionary procedures FND_MESSAGE.SET_NAME and FND_MESSAGE.SET_TOKEN to set the message name and tokens on the message dictionary stack. They can then call FND_MSG_PUB.Add to fetch the messages off the message dictionary stack and add it to the API message list.

To get a message from the API message list, API callers can use the procedure FND_MSG_PUB.Get. This procedure operates in five modes:

First Gets the first message in the API message list

Next Gets the next message in the API message list

Last Gets the last message in the API message list

Previous Gets the previous message in the API message list

Specific Gets a specific message from the API message list

For overall better performance and reduction in the number of calls a program needs to make in order to execute an API, it is recommended that APIs provide their callers with the following information:

- message count
- message data

The message count holds the number of messages in the API message list. If this number is one, then the message data holds the message in an encoded format.

```
x_msg_count   OUT   NUMBER
x_msg_data    OUT   VARCHAR2
```

Example:

```
  PROCEDURE allocate_line (
      p_material_details   IN       gme_material_details%ROWTYPE
     ,p_alloc_type         IN       NUMBER DEFAULT 9
     ,p_api_version        IN       NUMBER := gme_api_pub.api_version
     ,p_validation_level   IN       NUMBER := gme_api_pub.max_errors
     ,p_init_msg_list      IN       BOOLEAN := FALSE
     ,p_commit             IN       BOOLEAN := FALSE
     ,x_message_count      OUT      NUMBER
     ,x_message_list       OUT      VARCHAR2
     ,x_return_status      OUT      VARCHAR2
     ,p_del_exist_alloc    IN       BOOLEAN := FALSE) IS
```

```
      l_api_name   CONSTANT VARCHAR2 (30)                  := 'ALLOCATE_LINE';
      l_material_details    gme_material_details%ROWTYPE;
      l_batch_hdr               GME_BATCH_HEADER%ROWTYPE;
   BEGIN
      gme_debug.log_initialize('AllocateLine');
      -- Initialize message list and count if needed
      IF p_init_msg_list THEN
         fnd_msg_pub.initialize;
         gme_api_pub.error_count := 0;
      END IF;

      IF NOT gme_api_pub.setup_done THEN
         gme_api_pub.setup_done := gme_api_pub.setup;

         IF NOT gme_api_pub.setup_done THEN
            RAISE FND_API.g_exc_error;
         END IF;
      END IF;

      -- Make sure we are call compatible
      IF NOT FND_API.compatible_api_call (
                gme_api_pub.api_version
               ,p_api_version
               ,l_api_name
               ,'gme_api_allocate_line') THEN
         RAISE FND_API.g_exc_error;
         gme_api_pub.log_message (l_package_name||'.'||l_api_
name||':'||'INVALID_VERSION');
      END IF;

      gme_api_grp.set_timestamp;
      l_material_details := p_material_details;
      gme_api_allocate_line_pvt.allocate_line (
         p_gme_material_details => p_material_details
        ,p_alloc_type   => p_alloc_type
        ,x_gme_material_details => l_material_details
        ,x_return_status => x_return_status
        ,p_del_exist_alloc => p_del_exist_alloc);

      gme_debug.put_line(l_package_name||'.'||l_api_name||':'||'Return status
from private allocate_line is '||x_return_status);
      IF x_return_status = FND_API.G_RET_STS_SUCCESS THEN
         IF p_commit = TRUE THEN
            gme_debug.put_line (l_package_name||'.'||l_api_name||':'||'Calling
Save_batch');
            l_batch_hdr.batch_id := p_material_details.batch_id;
            save_batch(l_batch_hdr, x_return_status);
            IF x_return_status = FND_API.G_RET_STS_SUCCESS THEN
               COMMIT;
            ELSE
               gme_api_pub.log_message ('BATCH_SAVE_FAILED');
               RAISE FND_API.g_exc_error;
            END IF;
         END IF;
         IF error_count = 0 THEN
            log_message ('GME_API_LINE_ALLOCATED');
         END IF;
      END IF;
      gme_debug.put_line('Completed '||l_api_name ||' at '||to_
char(sysdate,'MM/DD/YYYY HH24:MI:SS'));


      fnd_msg_pub.count_and_get (
         p_count => x_message_count
        ,p_data => x_message_list);
   EXCEPTION
      WHEN FND_API.g_exc_error THEN
         x_return_status := FND_API.G_RET_STS_ERROR;
         fnd_msg_pub.count_and_get (
            p_encoded => FND_API.g_false
           ,p_count => x_message_count
           ,p_data => x_message_list);
      WHEN FND_API.g_exc_unexpected_error THEN
         x_return_status := FND_API.g_ret_sts_unexp_error;
         fnd_msg_pub.count_and_get (
            p_encoded => FND_API.g_false
           ,p_count => x_message_count
           ,p_data => x_message_list);
      WHEN OTHERS THEN
         x_return_status := FND_API.g_ret_sts_unexp_error;
         fnd_msg_pub.add_exc_msg (l_package_name, l_api_name);
         fnd_msg_pub.count_and_get (
            p_encoded => FND_API.g_false
           ,p_count => x_message_count
           ,p_data => x_message_list);
   END allocate_line;
```

Interpreting Error Conditions

The parameter x_return_status indicates whether the API was successful or failed. The values are:

- S for success
- E for error
- U for unknown or unexpected status

Understanding Error Messages

These error messages are output to the stored procedure message file, and can be monitored through the return x_msg_count. In conjunction with the x_return_status, this can be used to monitor the success or failure of the procedure call.

Displaying Errors in Languages Other than English

Language translation of error messages is determined by the environment variable NLS_LANGUAGE. If the message is not found in the required language, then the message is retrieved in US English.

The following is a complete list of Process Execution API Error Messages. Note that a message that is preceded with Warning is not an API error, just a warning, and a message preceded with Error is an API error.

Any uppercase word preceded by an apersand (&) is a token, or placeholder, for an actual value that is populated at runtime.

Message Text	**Message Name**
Batch is already saved.	BATCH_ALREADY_SAVED
Batch status is not updated.	BATCH_STATUS_NOT_UPDATED
All material lines that contribute to step quantity can be associated with a step when automatically.	GME_ALL_MATL_STEP_NOT_ASSOC
Quantity entered creates a negative actual quantity.	GME_API_ACTUAL_CANT_GO_NEG
Actual costing has been run for this batch.	GME_API_ACTUAL_COST_DONE_ERROR
Autoallocation was successful.	GME_API_BATCH_ALLOCATED
Batch has been cancelled.	GME_API_BATCH_CANCELLED

Message Text	Message Name
Batch has been completed.	GME_API_BATCH_CERTIFIED
Batch has been closed.	GME_API_BATCH_CLOSED
Batch has been created.	GME_API_BATCH_CREATED
Batch cannot be found with information supplied.	GME_API_BATCH_FETCH_ERROR
Batch Header Update Error.	GME_API_BATCH_HEADER_UPD_ERROR
Batch material lines cannot be locked.	GME_API_BATCH_LINES_LOCKED
Batch has been released.	GME_API_BATCH_RELEASED
Batch has been reopened.	GME_API_BATCH_REOPENED
Batch has been rerouted.	GME_API_BATCH_REROUTED
Cannot load the batch step with step details supplied.	GME_API_BATCH_STEP_FETCH_ERR
Cannot lock the batch step.	GME_API_BATCH_STEP_LINE_LCK
Error Reopening Batch Steps.	GME_API_BATCH_STEP_REOPEN_ERR
Cannot update the batch step.	GME_API_BATCH_STEP_UPD_ERR
Batch has been reverted to WIP status.	GME_API_BATCH_UNCERTIFIED
Batch has been unreleased.	GME_API_BATCH_UNRELEASED
Batch step is already closed.	GME_API_CLOSE_STEP_STATUS
Completion date is outside the valid range.	GME_API_CMPLT_OUT_VALIDITY
Cannot reopen the batch. The costing period for the calendar is closed.	GME_API_COST_PERIOD_CLOSED
Batch has been costed and period is open. Please run the actual cost process again.	GME_API_COST_PERIOD_OPEN
Unable to create the batch for the requested product quantity.	GME_API_CREATE_BY_PROD_FAIL
Cannot close batch step. Previous steps are not closed.	GME_API_DEP_STEP_N_CLS
Cannot reopen the batch step. One or more dependent steps are closed. You must reopen these steps before you can open this batch step.	GME_API_DEP_STEP_REOPEN

Message Text	Message Name
Cannot revert batch step to WIP. Succeeding step must have a status of pending or WIP.	GME_API_DEP_STEP_WIP
Cannot reopen the batch. It has been posted to subledger.	GME_API_GL_POSTED
Pending allocations exist for manual release items.	GME_API_INCOMP_MANUAL_TRANS
Batch step cannot be completed. Batch must have a status of WIP to complete the batch step.	GME_API_INV_BATCH_CERT_STEP
Cannot close batch step. Batch must have a status of WIP or completed.	GME_API_INV_BATCH_CLOSE_STEP
Batch details cannot be edited. Current step status is invalid.	GME_API_INV_BATCH_EDIT_STEP
Batch step cannot be released. Batch must have a status of WIP to release the batch step.	GME_API_INV_BATCH_REL_STEP
Batch cannot be rescheduled. Batch must have a status of pending or WIP to be rescheduled.	GME_API_INV_BATCH_RESCHED
Cannot perform incremental backflushing. Batch must have WIP or completed status.	GME_API_INV_BATCH_STATUS_PC
Cannot reopened batch step. Batch must have a status of WIP or Completed.	GME_API_INV_BATCH_STATUS_REOP
You cannot reopen firm planned order.	GME_API_INV_BATCH_TYPE
Batch step cannot be reverted to WIP. Batch must have a status of WIP.	GME_API_INV_BATCH_UNCERT_STEP
Cannot unrelease batch step. Batch step must have a status of WIP to be unreleased.	GME_API_INV_BATCH_UNRELE_STEP
Current step cannot be unreleased. The steps that follow this step must have a status of pending.	GME_API_INV_DEP_STEP_UNRELE
Planned quantity must be greater than zero to perform incremental backflushing.	GME_API_INV_PLAN_QTY_PC

Message Text	Message Name
Cannot drive incremental backflushing from an automatic release item, or from an automatic by step.	GME_API_INV_RELEASE_TYPE
Batch step cannot be completed. Batch step must have a status of pending or WIP to be completed.	GME_API_INV_STAT_STEP_CERT
Batch step cannot be closed. Batch step must have a status of completed to be closed.	GME_API_INV_STAT_STEP_CLS
Batch step cannot be cancelled. Batch step must have a status of pending to be cancelled.	GME_API_INV_STAT_STEP_CNCL
Step details cannot be updated. Current step status is invalid.	GME_API_INV_STAT_STEP_EDIT
Batch step cannot be released. Batch step must have a status of pending to be released.	GME_API_INV_STAT_STEP_REL
Batch step cannot be rescheduled. Batch step must have a status of pending.	GME_API_INV_STEP_STAT_RESCH
Batch step cannot be reverted to WIP. Step must have a status of completed.	GME_API_INV_STEP_STAT_UNCERT
Cannot unrelease batch step. Batch step must have a status of WIP to be unreleased.	GME_API_INV_STEP_STAT_UNRELE
Cannot reopen batch step. Batch step must have a status of closed to be reopened.	GME_API_INV_STEP_STATUS_REOP
Batch cannot be cancelled. Batch must have a status of pending to be cancelled.	GME_API_INVALID_BATCH_CANCEL
Batch cannot be completed. Batch must have a status of WIP to be completed.	GME_API_INVALID_BATCH_CERTIFY
Batch cannot be released. Batch must have a status of pending to be released.	GME_API_INVALID_BATCH_REL
Batch cannot be rerouted. Batch must have a status of pending.	GME_API_INVALID_BATCH_REROUTE
Batch cannot be reverted to WIP. Batch must have a status of completed.	GME_API_INVALID_BATCH_UNCERT

Message Text	Message Name
Batch cannot be unreleased. Batch must have a status of WIP to be unreleased.	GME_API_INVALID_BATCH_UNREL
Cannot create the batch. Formula supplied is not valid.	GME_API_INVALID_FORMULA
QTY_TYPE parameter must be 0 for incremental, 1 for percentage, or 2 for actual.	GME_API_INVALID_INCR_TYPE
Validity rule is not valid.	GME_API_INVALID_RULE
Cannot unrelease batch. All the steps must have a status of WIP or pending.	GME_API_INVALID_STEP_UNREL
Validity rule was not found.	GME_API_INVALID_VALIDITY
Autoallocation was successful.	GME_API_LINE_ALLOCATED
Cannot lock inventory.	GME_API_LOCKING_FAILURE
This batch is marked for deletion.	GME_API_MARKED_FOR_DELETION
Batch material line cannot be found with information supplied.	GME_API_MATL_DTL_FETCH_ERROR
Material detail line cannot be created.	GME_API_MATL_DTL_SETUP_FAILURE
Batch material detail line cannot be updated.	GME_API_MATL_DTL_UPD_ERROR
Incremental backflush was successful.	GME_API_PARTIAL_CERTIFIED
Quantity entered must be greater than zero.	GME_API_QTY_CANT_BE_ZERO
Actual start date of the step cannot be in the future.	GME_API_REL_STEP_FUTURE_DATE
You must enter a date to reschedule a batch.	GME_API_RESCH_NO_DATES_PASSED
Dates entered for rescheduling the batch step are the same as those currently entered for the step.	GME_API_RESCH_STEP_NO_DATE_CHG
Dates entered in for rescheduling the batch step cannot be NULL.	GME_API_RESCH_STEP_NO_DATES
Process quantities of &RESOURCES fall below the minimum capacity.	GME_API_RSRC_QTY_BELOW_CAP

Message Text	Message Name
The validity rule selected is the same as the current validity rule.	GME_API_SAME_VALIDITY_RULE
Setup failed.	GME_API_SETUP_FAILURE
Start date is outside valid range.	GME_API_START_OUT_VALIDITY
Step is deleted.	GME_API_STEP_DELETE
Batch step has been reopened.	GME_API_STEP_REOPENED
Batch steps cannot be updated.	GME_API_STEP_UPD_ERROR
Cannot reopen the batch. Batch transactions have been purged.	GME_API_TRANSACTIONS_PURGED
Unable to retrieve the constant &CONSTANT_NAME.	GME_API_UNABLE_TO_GET_CONSTANT
All material lines are not allocated.	GME_API_UNALLOC_MATERIALS
The validity rule selected has a different formula. To reroute batch, you must select a validity rule that has the same formula.	GME_API_VALIDITY_DIFF_FORM
The warehouse &WHSE_CODE is closed for the date &TRANS_DATE. Please reenter the date to proceed.	GME_API_WHSE_CLOSED
Plant Warehouse relationship is not defined for all items in the batch.	GME_API_WSHE_LOOKUP_FAILURE
Error retrieving the recipe data.	GME_BAD_RECIPE_RETRIEVAL
Batch is locked by another user.	GME_BATCH_IN_USE
The batch &BATCH_NO does not have any inventory transactions.	GME_BATCH_NON_INVENTORY
Batch step completed successfully.	GME_BATCH_STEP_CERTIFIED
Batch step has been closed.	GME_BATCH_STEP_CLOSED
Batch step not found for step id &STEP_ ID.	GME_BATCH_STEP_NOT_FOUND
Batch step has been released.	GME_BATCH_STEP_RELEASED
Batch step has been successfully reverted to WIP.	GME_BATCH_STEP_UNCERTIFIED
Batch step has been unreleased.	GME_BATCH_STEP_UNRELEASED

Message Text	Message Name
Charges cannot be calculated for step &STEP_NO.	GME_CALC_CHARGE_CONV_ERROR
Transaction date &TRANS_DATE falls in closed period for Work In Progress Warehouse.	GME_DATE_IN_CLSD_PRD
The planned start and planned completion dates are not within the validity rule dates.	GME_DATES_EXCEED_VALDTY_RULE
Manual document ordering is set for this plant. Please supply document number.	GME_DOC_NUM_NOT_PASSED
This batch already exists. Please enter another batch number.	GME_DUP_BATCH
Cannot change the formula ID for the current batch.	GME_FORMID_CHG_NOT_ALLOWED
Cannot create firm planned order for Lab Organization.	GME_FPO_NO_CREATE
Insufficient validity rules data supplied.	GME_INSUF_VAL_RULE
The cost allocation must be zero for inserting the material line.	GME_INV_COST_ALLOC_INS
Invalid option for updating the cost allocation.	GME_INV_COST_UPD_OPT
Cannot insert the material line. The formula line ID must be NULL to insert a material line.	GME_INV_FORMULALINE
You cannot drive Inventory quantity negative.	GME_INV_NEG_NOT_ALLOW
The yield type of the phantom product cannot be changed.	GME_INV_PHANT_UPD_REL
Invalid value for phantom type.	GME_INV_PHANTOM_TYPE
Inventory shortages found for the batch.	GME_INV_SHORT_EXISTS
Invalid batch status for updating the phantom type.	GME_INV_STAT_UPD_PHANTOM_TYPE
Invalid batch status for updating planned quantity.	GME_INV_STAT_UPD_PLAN_QTY
Invalid batch status for updating the release type.	GME_INV_STAT_UPD_REL

Message Text	Message Name
Invalid batch status for updating the scrap factor.	GME_INV_STAT_UPD_SCRAP
Invalid batch status for updating the WIP planned quantity.	GME_INV_STAT_UPD_WIP_PLAN
Invalid batch status for inserting a step.	GME_INV_STATUS_INSERT_STEP
Cannot edit resource transaction if step status is released and Automatic step quantity calculation is selected.	GME_INV_STEP_STATUS_ASQC
The actual start date or the actual completion date is not within the validity rule dates.	GME_INVAL_VAL_RULE_DATES
Invalid API version.	GME_INVALID_API_VERSION
Batch status is not valid for &PROCESS.	GME_INVALID_BATCH_STATUS
Invalid batch type. The batch type must be 0 or 10.	GME_INVALID_BATCH_TYPE
&DATE1 cannot be earlier than &DATE2.	GME_INVALID_DATE_RANGE
Instance ID &INSTANCE_ID is not found in GMP_RESOURCE_ INSTANCES table.	GME_INVALID_INSTANCE_ID
Instance number &INSTANCE_NO is not found in GMP_RESOURCE_ INSTANCES table.	GME_INVALID_INSTANCE_NO
Invalid line number.	GME_INVALID_LINE_NO
Invalid value for Line Type.	GME_INVALID_LINE_TYPE
Invalid value for planned quantity.	GME_INVALID_PLAN_QTY
Organization code supplied is not a manufacturing plant or laboratory.	GME_INVALID_PLANT
Invalid value for scale type.	GME_INVALID_SCALE_TYPE
Invalid value for the scrap factor.	GME_INVALID_SCRAP_FACTOR
Invalid value for the field text_code.	GME_INVALID_TEXT_CODE
Invalid value for WIP planned quantity.	GME_INVALID_WIP_PLAN_QTY
Updating the item ID is not allowed for the current batch.	GME_ITEMID_CHG_NOT_ALLOWED

Message Text	Message Name
Updating the line number is not allowed for the current batch.	GME_LINE_NO_CHG_NOT_ALLOWED
Updating the line type is not allowed for the current batch.	GME_LINE_TYPE_CHG_NOT_ALLOWED
Invalid batch status for deleting the material detail line.	GME_MATL_DEL_INV_STAT
Invalid batch status for inserting the material detail line.	GME_MATL_INS_INV_STAT
Invalid batch status for updating the material detail line.	GME_MATL_UPD_INV_STAT
Maximum step capacity is not defined for &MASS &VOLUME unit of measure types.	GME_MAX_CAP_NOT_MASS_VOL_ERR
The automatic step quantity calculation requires a minimum of one material line that contributes to step quantity.	GME_MISS_LINES_CONT_STEP
Negative resource usage is not allowed.	GME_NEG_USAGE_NT_ALWD
Activities are not defined for step &STEP_ NO	GME_NO_ACTIVITIES
No data found while retrieving the row from table &TABLE_NAME.	GME_NO_DATA_FOUND
Cannot delete a phantom ingredient.	GME_NO_DEL_PHANT_ING
Cannot delete a phantom product.	GME_NO_DEL_PHANT_PROD
No keys specified to identify the rows in the table &TABLE_NAME.	GME_NO_KEYS
No operation defined. Please define the operation.	GME_NO_OPRN_DEFINED
No Plant code defined.	GME_NO_PLANT_CODE_DEFINED
Resources are not defined for activity &ACTIVITY	GME_NO_RESOURCES
Step was not passed to the retrieve unallocated items routine.	GME_NO_STEP_FETCH_UNALL
At least one ingredient and one product are required.	GME_ONE_ING_PROD_REQD

Message Text	Message Name
At least one step required.	GME_ONE_STEP_REQD
Cannot update the material line to Not a Phantom, since a phantom batch already exists.	GME_PHANT_BATCH_EXISTS
Enter the value for Planned Resource Usage.	GME_PLAN_RSRC_REQD
The primary product line of a batch cannot be deleted.	GME_PRIM_PROD_DEL_ERR
The quantity entered does not match the material allocated. To enter a different quantity, click Line Allocations button.	GME_QTY_UNMATCH_ALLOC
&KEY Record of &TABLE_NAME is locked by another user.	GME_RECORD_LOCKED
Resource Actual Dates must be within the parent actual dates (&START_DATE - &END_DATE).	GME_RSRC_ACTUAL_DATE
Enter the value for Planned Resource Count.	GME_RSRC_COUNT_REQD
Resource &RESOURCE was not found for activity &ACTIVITY.	GME_RSRC_NOT_FOUND
Resource Planned Dates are not within the range of activity planned dates (&START_ DATE - &END_DATE)	GME_RSRC_PLAN_DATE
Enter the value for Planned Resource Quantity.	GME_RSRC_QTY_REQD
Cannot compute resource usage as GMP: UOM for Hours - &SY_UOM cannot be converted to resource usage unit of measure &RSRC_USG_UOM.	GME_RSRC_USG_NT_CNV_SYUOM
Batch step resource ID &BATCHSTEP_RSRC_ID was not found in GME_BATCH_STEP_ RESOURCES table.	GME_RSRCID_NOT_FOUND
Resource transaction cannot be added for firm planned order.	GME_RTXN_FOR_FPO_NT_ALWD

Message Text	Message Name
Resource transaction cannot be added for a lab batch that cannot have its inventory updated.	GME_RTXN_FOR_UPDINV_NT_ALWD
Enter the value for Planned Start Date.	GME_START_DATE_REQD
Activity &ACTIVITY is not defined for step &STEP_NO.	GME_STEP_ACTIVITY_NOT_FOUND
Batch step cannot be deleted. Batch must have a status of pending to delete batch step.	GME_STEP_NOT_PENDING
Cannot reschedule steps. A step timing conflict was detected.	GME_STEP_OVERLAP_ERROR
Planned step quantity cannot be less than zero.	GME_STEP_QTY_LTHAN_ZERO
Enter the value for &FIELDREQ.	GME_STEP_REQD
Required field missing for the transaction row.	GME_TRANS_REQD_FLD_MISSING
Dates are not updated for allocations	GME_TRANS_UPD_DATE
Unexpected &ERROR occurred.	GME_UNEXPECTED_ERROR
ERROR in &PROC - Input parameters are missing.	INPUT_PARMS_MISS
Firm planned order steps cannot be closed.	INVALID_BATCH_TYPE_CLS_STEP
Default lot cannot be found.	NO_DEFAULT_LOT_FOUND
Invalid Batch Step Number	PC_INV_BATCHSTEP_NO
Starting Date cannot be greater than the End Date.	PM_BADSTARTDATE
No Transactions found.	PM_DEFAULT_TRANS_LOST
This indivisible lot has already been allocated elsewhere.	PM_INDIV_ALREADY_ALLOCD
Invalid item.	PM_INVALID_ITEM
Invalid action for phantom batches.	PM_INVALID_PHANTOM_ACTION
Lot is indivisible. Allocate entire available qty or select another lot/sublot.	PM_LOT_INDIV

Message Text	Message Name
Missing or Invalid Allocation Class.	PM_NO_ALLOC_CLASS
This is not a Phantom Ingredient.	PM_NOTAPHANTOM
Item is not lot/location controlled.	PM_NOTLOT_LOCT
Phantom Batch already exists for this Item.	PM_PHANTOM_EXISTS
Unexploded Phantoms Exist.	PM_UNEXPLODED_PHANTOMS
Setup Error.	SETUP_ERROR
Unable To Load UOM.	UNABLE_TO_LOAD_UOM

B

Listing of GMEPAPIS.pls

```
REM dbdrv: sql ~PROD ~PATH ~FILE none none none package &phase=pls \
REM dbdrv: checkfile:~PROD:~PATH:~FILE
SET VERIFY OFF
WHENEVER SQLERROR EXIT FAILURE ROLLBACK;
REM *******************************************************************
REM *                                                                 *
REM * FILE:    GMEPAPIS.pls                                           *
REM * PURPOSE: Package Specification for the GME PUBLIC API routines  *
REM * AUTHOR:  Paul J Schofield, OPM Development                      *
REM * DATE:    February 1st 2001                                      *
REM * HISTORY:                                                        *
REM * ========                                                        *
REM * 01FEB01  P.J.Schofield                                          *
REM *          Created                                                *
REM * 21MAR01  Thomas Daniel                                          *
REM *          Added unallocated materials record type definition     *
REM * 07MAY01  Thomas Daniel                                          *
REM *        Added Certify_Step, Uncertify_Step, Theoretical_Yield,   *
REM *          Reroute_Batch and Resource Consolidation routines      *
REM *          Created                                                *
REM * 08MAY01  Bharati Satpute                                        *
REM *          Added Close Batch, Close Steps routines Created        *
REM * 15MAY01  Shikha Nagar                                           *
REM *          Added Partial Certification Routines                   *
REM * 23MAY01  Thomas Daniel                                          *
REM *          Added insert step routine                              *
REM * 23MAY01  Bharati Satpute                                        *
REM *           Added Unrelease step routine                          *
REM * 24MAY01  Thomas Daniel                                          *
REM *          Added Reschedule step routine                          *
REM * 31MAY01  Olivier Daboval                                        *
REM *          Added Reopen_batch routine                             *
REM * 06JUN01  Olivier Daboval                                               *
REM *          Added reopen_step routine                                     *
REM * 06JUN01  Shrikant Nene                                                 *
REM *          Changed gme_text_tbl to gme_text_table                        *
REM * 18JUN01  Shrikant Nene                                                 *
REM *          Added release_step routine                                    *
REM * 14AUG01  Thomas Daniel                                                 *
REM *          Added call to material line APIs.                             *
REM * 21AUG01  Pawan Kumar                                                   *
REM *          Added call to convert_fpo APIs.                               *
REM * 27AUG01  Pawan Kumar                                                   *
REM *          Added call to scale_batch and theoretical_yield_batch.        *
REM * 07JAN02  BUG 2159185 - Added parameter                                 *
REM *          p_update_batch to determine if the batch planned end          *
REM *          date should be updated if the rescheduling goes past the      *
REM *          batch's planned end date or if an error message should        *
REM *          be returned.  Modified procedure reschedule_step.             *
REM * 25APR02  BUG 2342448 - Thomas Daniel                                   *
REM *          Removed the word batch in the name of the procedures          *
REM *          certify_step and uncertify_step. Also removed the out         *
REM *          parameter from the delete_material_line procedure.            *
REM * 03MAY02  BUG 2359928 - Thomas Daniel                                   *
REM *          Added the new insert_line_allocation API.                     *
REM * 21MAY02  BUG 2367604 - Thomas Daniel                                   *
REM *          Added the parameter p_delete_pending for close step.          *
REM **********************************************************************

/*************************************************************************
* This file contains the headers for the Process Execution (GME) APIs in *
* Oracle Process Manufacturing (OPM). Each procedure has a common set of *
* parameters to which API-specific parameters are appended.              *
*************************************************************************/
CREATE OR REPLACE PACKAGE gme_api_pub AS
/* $Header: GMEPAPIS.pls 115.42 2002/05/31 23:28:29 snene noship $ */

   api_version           CONSTANT NUMBER                              := 1;
   max_errors            CONSTANT NUMBER                              := 100;
   inv_short_err         CONSTANT VARCHAR2 (1)                        := 'V';
   unalloc_items_err     CONSTANT VARCHAR2 (1)                        := 'N';
   incomp_man_alloc_err  CONSTANT VARCHAR2 (1)                        := 'I';
   negative_inventory    CONSTANT VARCHAR2 (1)                        := 'G';
   setup_done                     BOOLEAN                             :=
FALSE;
   user_name                      VARCHAR2 (240);
   user_ident                     NUMBER;
   login_id                          NUMBER;
   default_lot                       VARCHAR2 (240);
   default_loct                      VARCHAR2 (240);
   yield_type                        VARCHAR2 (240);
   yield_uom                         VARCHAR2 (240);
   allow_neg_inv                     VARCHAR2 (240);
   step_control                      VARCHAR2 (240);
   error_count                       NUMBER                              DEFAULT
0;
   co_code                           VARCHAR2 (4);
   plant_code                        VARCHAR2 (4);
   copy_routing_text                 VARCHAR2 (4);
   copy_formula_text                 VARCHAR2 (4);
   TIMESTAMP                         DATE;
   release_type                      NUMBER;
   auto_release          CONSTANT NUMBER (5)                           := 0;
   manual_release        CONSTANT NUMBER (5)                           := 1;
   incremental_release   CONSTANT NUMBER (5)                           := 2;
   step_release          CONSTANT NUMBER (5)                           := 3;
   auto_auto_alloc       CONSTANT NUMBER (5)                           := 1;
   user_auto_alloc       CONSTANT NUMBER (5)                           := 0;
   epsilon                           NUMBER;
   batch_scale_factor                NUMBER;
   routing_scale_factor              NUMBER;
   warn_message                      VARCHAR2 (240);

   TYPE number_tab IS TABLE OF NUMBER
      INDEX BY BINARY_INTEGER;

   TYPE material_details_tab IS TABLE OF gme_material_details%ROWTYPE
      INDEX BY BINARY_INTEGER;

   TYPE item_masters_tab IS TABLE OF ic_item_mst%ROWTYPE
      INDEX BY BINARY_INTEGER;

   TYPE transactions_tab IS TABLE OF gme_inventory_txns_gtmp%ROWTYPE
      INDEX BY BINARY_INTEGER;

   TYPE real_inv_trans_tab IS TABLE OF ic_tran_pnd%ROWTYPE
      INDEX BY BINARY_INTEGER;

   TYPE inv_trans_rec_tab IS TABLE OF gmi_trans_engine_pub.ictran_rec
      INDEX BY BINARY_INTEGER;

   TYPE real_res_trans_tab IS TABLE OF gme_resource_txns%ROWTYPE
```

```
      INDEX BY BINARY_INTEGER;

   TYPE text_tab IS TABLE OF gme_text_table%ROWTYPE
      INDEX BY BINARY_INTEGER;

   TYPE resource_transactions_tab IS TABLE OF gme_resource_txns_gtmp%ROWTYPE
      INDEX BY BINARY_INTEGER;

   TYPE unallocated_items_tab IS TABLE OF gme_unallocated_items_gtmp%ROWTYPE
      INDEX BY BINARY_INTEGER;

   TYPE p_field IS RECORD (p_value VARCHAR2 (50));

   TYPE field_values_tab IS TABLE OF p_field
      INDEX BY BINARY_INTEGER;

   TYPE recipe_validity_rule_tab IS TABLE OF gmd_recipe_validity_rules%ROWTYPE
      INDEX BY BINARY_INTEGER;

   --Buffers for recipe upload
   routings                        gmd_recipe_fetch_pub.recipe_rout_tbl;
   routing_materials               gmd_recipe_fetch_pub.recipe_rout_matl_tbl;
   steps                           gmd_recipe_fetch_pub.recipe_step_tbl;
   step_dependencies               gmd_recipe_fetch_pub.routing_depd_tbl;
   activities                      gmd_recipe_fetch_pub.oprn_act_tbl;
   resources                       gmd_recipe_fetch_pub.oprn_resc_tbl;
   materials                       gmdfmval_pub.formula_detail_tbl;
   return_code                     NUMBER;
   recipe_id                       NUMBER;

   TYPE unallocated_materials_tab IS TABLE OF gme_unallocated_items_gtmp%ROWTYPE
      INDEX BY BINARY_INTEGER;


/*================================================================================
==
    Procedure
      log_message
    Description
      This procedure is used accross all the procedures to log a message to the
      message stack.
    Parameters
      p_mesage_code (R)    The message which is being put onto the stack.
      p_token1_name (R)    The name of the token1 in the message if any.
      p_token1_value (R)   The value of the token1 in the message if it exists.
      p_token2_name (R)    The name of the token2 in the message if any.
      p_token2_value (R)   The value of the token2 in the message if it exists.
      p_token3_name (R)    The name of the token3 in the message if any.
      p_token3_value (R)   The value of the token3 in the message if it exists.

================================================================================
*/

   PROCEDURE log_message (
      p_message_code   IN   VARCHAR2
     ,p_token1_name    IN   VARCHAR2 := NULL
     ,p_token1_value   IN   VARCHAR2 := NULL
     ,p_token2_name    IN   VARCHAR2 := NULL
     ,p_token2_value   IN   VARCHAR2 := NULL
     ,p_token3_name    IN   VARCHAR2 := NULL
     ,p_token3_value   IN   VARCHAR2 := NULL);


/*==============================================================================
==
    FUNCTION
      setup
    Description
      This function is used accross all the procedures to setup the profile
values
      and constants. This function returns FALSE if any of the constants or
profiles
      are not set properly.
    Return Values
      TRUE        If the setup is done successfully.
      FALSE       If their are any errors in the setup.

================================================================================
*/
   FUNCTION setup
      RETURN BOOLEAN;


/*==============================================================================
==
    Procedure
      save_batch
    Description
      This procedure is used to consolidate all the transactions from the
temporary
```

```
      tables and write them to the main tables.
    Parameters
      p_batch_header (R)        The batch header row to identify the batch
                                Following columns are used from this row.
                                batch_id  (R) OR
                                plant_code (R)
                                batch_type (R)
                                batch_no (R)
      x_return_status           outcome of the API call
                                S - Success
                                E - Error
                                U - Unexpected Error

================================================================================
*/

   PROCEDURE save_batch (
      p_batch_header    IN       gme_batch_header%ROWTYPE
     ,x_return_status   OUT      VARCHAR2);


/*==============================================================================
==
    Procedure
      create_batch
    Description
      This procedure creates batch, then check for Items failing allocation and
           inventory shortages.
    Parameters
      p_batch_header (R)        The batch header row to identify the batch
                                Following columns are used from this row.
                                plant_code  (R)
                                recipe_validity_rule_id (R)
                                batch_type (R)
                                update_inventory_ind (R)
                                batch_no (R In case of manual document ordering)
                                plan_start_date (O)
                                plan_cmplt_date (O)
                                due_date (O)
                                wip_whse_code (O)
      p_batch_size (R)         Batch Size (Total input, total output or product
quantity)
      p_batch_size_uom (R)      UOM for p_batch_size
      p_ignore_shortages (R)    Do not check for the inventory shortages
      p_creation_mode (R)       How the batch is created
```

```
                                RECIPE
                                PRODUCT
                                TOTAL_OUTPUT
                                TOTAL_INPUT
      p_recipe_id (O)           Recipe_id for which the batch is to be created.
      p_recipe_no (O)           Recipe number along with recipe_version for
which the
                                batch is to be created.
      p_recipe_version (O)      Version of the recipe for  which the batch is t
be created.
      p_product_no (O)          Item number for which the batch is to be
created.
      p_product_id (O)          Item ID for which the batch is to be created.
      p_ignore_qty_below_cap (O)Whether the batch is to be created or not, when
resource
                                quantity goes below minimum capacity of the
resource.
                                True  (Default)
                                False

      x_batch_header            The batch header that is returned, with all the
data
      x_unallocated_material    Table of materials, if auto allocation failed or
                                inventory shortage exists
      x_return_status           outcome of the API call
                                S - Success
                                E - Error
                                U - Unexpected Error
                                N - Items failed auto allocation
                                V - Inventory shortage exists

================================================================================
*/

   PROCEDURE create_batch (
      p_api_version            IN       NUMBER := gme_api_pub.api_version
     ,p_validation_level       IN       NUMBER := gme_api_pub.max_errors
     ,p_init_msg_list          IN       BOOLEAN := FALSE
     ,p_commit                 IN       BOOLEAN := FALSE
     ,x_message_count          OUT      NUMBER
     ,x_message_list           OUT      VARCHAR2
     ,x_return_status          OUT      VARCHAR2
     ,p_batch_header           IN       gme_batch_header%ROWTYPE
     ,x_batch_header           OUT      gme_batch_header%ROWTYPE
     ,p_batch_size             IN       NUMBER
     ,p_batch_size_uom         IN       VARCHAR2
     ,p_creation_mode          IN       VARCHAR2
     ,p_recipe_id              IN       NUMBER := NULL
     ,p_recipe_no              IN       VARCHAR2 := NULL
     ,p_recipe_version         IN       NUMBER := NULL
     ,p_product_no             IN       VARCHAR2 := NULL
     ,p_product_id             IN       NUMBER := NULL
     ,p_ignore_qty_below_cap   IN       BOOLEAN := TRUE
     ,p_ignore_shortages       IN       BOOLEAN
     ,x_unallocated_material   OUT      gme_api_pub.unallocated_materials_tab);


/*================================================================================
==
    Procedure
      create_phantom
    Description
      This procedure creates phantom batch based on the validity rule passsed

    Parameters
      p_material_details (R)    The material detail row to identify the material
                                Following columns are used from this row.
                                material_detail_id  (R)
      p_validity_rule_id (R)    validity rule to use for creating phantom batch
      p_ignore_shortages (R)    Do not check for the inventory shortages

      x_material_details        The material detail that is returned, with all
the data
      x_unallocated_material    Table of materials, if auto allocation failed or
                                inventory shortage exists
      x_return_status           outcome of the API call
                                S - Success
                                E - Error
                                U - Unexpected Error

================================================================================
*/

   PROCEDURE create_phantom (
      p_api_version            IN       NUMBER := gme_api_pub.api_version
     ,p_validation_level       IN       NUMBER := gme_api_pub.max_errors
     ,p_init_msg_list          IN       BOOLEAN := FALSE
     ,p_commit                 IN       BOOLEAN := FALSE
     ,x_message_count          OUT      NUMBER
     ,x_message_list           OUT      VARCHAR2
     ,x_return_status          OUT      VARCHAR2
     ,p_material_detail        IN       gme_material_details%ROWTYPE
     ,x_material_detail        OUT      gme_material_details%ROWTYPE
     ,p_validity_rule_id       IN       NUMBER
     ,p_ignore_shortages       IN       BOOLEAN
     ,x_unallocated_material   OUT      gme_api_pub.unallocated_materials_tab);


/*================================================================================
==
    Procedure
      release_batch
    Description
      This procedure releases batch and all the phantom batches
       Before releaseing the batch, it tries to auto-allocate batch, check
       for inventory shortages.

    Parameters
      p_batch_header (R)        The batch header row to identify the batch
                                Following columns are used from this row.
                                batch_id  (R)
                                actual_start_date (O)
      p_ignore_shortages (R)    Do not check for the inventory shortages
      p_ignore_unalloc (R)      Do not check for the item requiring allocations

      x_batch_header            The batch header that is returned, with all the
data
      x_unallocated_material    Table of materials, if auto allocation failed or
                                    inventory shortage exists
      x_return_status           outcome of the API call
                                S - Success
                                E - Error
                                U - Unexpected Error
                                N - Items require allocation
                                V - Inventory shortage exists

================================================================================
*/

   PROCEDURE release_batch (
      p_api_version            IN       NUMBER := gme_api_pub.api_version
     ,p_validation_level       IN       NUMBER := gme_api_pub.max_errors
     ,p_init_msg_list          IN       BOOLEAN := FALSE
     ,p_commit                 IN       BOOLEAN := FALSE
     ,x_message_count          OUT      NUMBER
     ,x_message_list           OUT      VARCHAR2
     ,x_return_status          OUT      VARCHAR2
     ,p_batch_header           IN       gme_batch_header%ROWTYPE
     ,x_batch_header           OUT      gme_batch_header%ROWTYPE
     ,p_ignore_shortages       IN       BOOLEAN
     ,x_unallocated_material   OUT      gme_api_pub.unallocated_materials_tab
     ,p_ignore_unalloc         IN       BOOLEAN DEFAULT FALSE);


/*================================================================================
==
    Procedure
      release_step
    Description
      This procedure releases step. If the batch is not in WIP state already and
the
       GME: Step control batch status is set to true, this procedure will call
       release_batch.
       Before releaseing the step, it tries to auto-allocate batch, check
       for inventory shortages.

    Parameters
      p_batch_step (R)          The batch step row to identify the step
                                Following columns are used from this row.
                                batchstep_id  (R)
                                actual_start_date (O)
      p_ignore_shortages (R)    Do not check for the inventory shortages
      p_ignore_unalloc (R)      Do not check for the item requiring allocations

      x_batch_step              The batch header that is returned, with all the
data
      x_unallocated_material    Table of materials, if auto allocation failed or
                                    inventory shortage exists
      x_return_status           outcome of the API call
                                S - Success
                                E - Error
                                U - Unexpected Error
                                N - Items require allocation
                                V - Inventory shortage exists

================================================================================
*/

   PROCEDURE release_step (
      p_api_version            IN       NUMBER := gme_api_pub.api_version
     ,p_validation_level       IN       NUMBER := gme_api_pub.max_errors
     ,p_init_msg_list          IN       BOOLEAN := FALSE
     ,p_commit                 IN       BOOLEAN := FALSE
     ,p_batch_step             IN       gme_batch_steps%ROWTYPE
     ,x_message_count          OUT      NUMBER
     ,x_message_list           OUT      VARCHAR2
     ,x_return_status          OUT      VARCHAR2
     ,x_batch_step             OUT      gme_batch_steps%ROWTYPE
     ,x_unallocated_material   OUT      unallocated_materials_tab
     ,p_ignore_shortages       IN       BOOLEAN := FALSE
     ,p_ignore_unalloc         IN       BOOLEAN := FALSE);


/*================================================================================
==
    Procedure
      unrelease_batch
    Description
      This procedure unreleases batch and all the phantom batches



    Parameters
      p_batch_header (R)        The batch header row to identify the batch
                                Following columns are used from this row.
                                batch_id  (R)
      p_preserve_allocations (R)Do not delete pending allocations

      x_batch_header            The batch header that is returned, with all the
data
      x_return_status           outcome of the API call
                                S - Success
                                E - Error
                                U - Unexpected Error

================================================================================
*/

   PROCEDURE unrelease_batch (
      p_api_version            IN       NUMBER := gme_api_pub.api_version
     ,p_validation_level       IN       NUMBER := gme_api_pub.max_errors
     ,p_init_msg_list          IN       BOOLEAN := FALSE
     ,p_commit                 IN       BOOLEAN := FALSE
     ,x_message_count          OUT      NUMBER
     ,x_message_list           OUT      VARCHAR2
     ,x_return_status          OUT      VARCHAR2
     ,p_batch_header           IN       gme_batch_header%ROWTYPE
     ,x_batch_header           OUT      gme_batch_header%ROWTYPE
     ,p_preserve_allocations   IN       BOOLEAN);


/*================================================================================
==
    Procedure
      unrelease_step
    Description
      This procedure unreleases step.

    Parameters
      p_batch_step (R)          The batch step row to identify the step
                                Following columns are used from this row.
                                batchstep_id  (R)
      p_preserve_allocations (R)Do not delete pending allocations

      x_batch_step              The batch header that is returned, with all the
data
      x_return_status           outcome of the API call
                                S - Success
                                E - Error
                                U - Unexpected Error

================================================================================
*/

   PROCEDURE unrelease_step (
      p_api_version            IN       NUMBER := gme_api_pub.api_version
     ,p_validation_level       IN       NUMBER := gme_api_pub.max_errors
     ,p_init_msg_list          IN       BOOLEAN := FALSE
     ,p_commit                 IN       BOOLEAN := FALSE
     ,x_message_count          OUT      NUMBER
     ,x_message_list           OUT      VARCHAR2
     ,x_return_status          OUT      VARCHAR2
     ,p_batch_step             IN       gme_batch_steps%ROWTYPE
     ,x_batch_step             OUT      gme_batch_steps%ROWTYPE
     ,p_preserve_allocations   IN       BOOLEAN);


/*================================================================================
==
    Procedure
      certify_batch
    Description
      This procedure completes (certifies) batch and all the phantom batches.
       It also certifies all the steps.
       While completing the batch, checks for inventory shortages.

    Parameters
      p_batch_header (R)        The batch header row to identify the batch
                                Following columns are used from this row.
                                batch_id  (R)
                                actual_start_date (O) (In case of direct
completion)
                                actual_cmplt_date (O)
      p_ignore_shortages (R)    Do not check for the inventory shortages
      p_del_incomplete_manual (R) Delete incomplete manual transactions

      x_batch_header            The batch header that is returned, with all the
data
      x_unallocated_material    Table of materials, if inventory shortage
exists, or
                                 incomplete manual transactions exist
      x_return_status           outcome of the API call
                                S - Success
                                E - Error
                                U - Unexpected Error
                                N - Items require allocations
                                V - Inventory shortage exists
                                I - Incomplete manual transactions exists

================================================================================
*/

   PROCEDURE certify_batch (
      p_api_version             IN       NUMBER := gme_api_pub.api_version
     ,p_validation_level        IN       NUMBER := gme_api_pub.max_errors
     ,p_init_msg_list           IN       BOOLEAN := FALSE
     ,p_commit                  IN       BOOLEAN := FALSE
     ,x_message_count           OUT      NUMBER
     ,x_message_list            OUT      VARCHAR2
     ,x_return_status           OUT      VARCHAR2
     ,p_del_incomplete_manual   IN       BOOLEAN := FALSE
     ,p_ignore_shortages        IN       BOOLEAN := FALSE
     ,p_batch_header            IN       gme_batch_header%ROWTYPE
     ,x_batch_header            OUT      gme_batch_header%ROWTYPE
     ,x_unallocated_material    OUT      unallocated_materials_tab);
```

```
/*================================================================================
==
    Procedure
      certify_step
    Description
      This procedure completes (certifies) step. If this is the last step and
       GME: Step controls Batch status is set then It also calls complete_batch.
       While completing the step, checks for inventory shortages.

    Parameters
      p_batch_step (R)          The batch step row to identify the step
                                Following columns are used from this row.
                                batchstep_id  (R)
                                actual_start_date (O) (In case of direct
completion)
                                actual_cmplt_date (O)
      p_ignore_shortages (R)    Do not check for the inventory shortages
      p_del_incomplete_manual (R) Delete incomplete manual transactions

      x_batch_step              The batch step that is returned, with all the
data
      x_unallocated_material    Table of materials, if inventory shortage
exists, or
                                 incomplete manual transactions exist
      x_return_status           outcome of the API call
                                S - Success
                                E - Error
                                U - Unexpected Error
                                N - Items require allocations
                                V - Inventory shortage exists
                                I - Incomplete manual transactions exists

================================================================================
*/

   PROCEDURE certify_step (
      p_api_version             IN       NUMBER := gme_api_pub.api_version
     ,p_validation_level        IN       NUMBER := gme_api_pub.max_errors
     ,p_init_msg_list           IN       BOOLEAN := FALSE
     ,p_commit                  IN       BOOLEAN := FALSE
     ,x_message_count           OUT      NUMBER
     ,x_message_list            OUT      VARCHAR2
     ,x_return_status           OUT      VARCHAR2
     ,p_batch_step              IN       gme_batch_steps%ROWTYPE
     ,x_batch_step              OUT      gme_batch_steps%ROWTYPE
     ,x_unallocated_material    OUT      unallocated_materials_tab
     ,p_del_incomplete_manual   IN       BOOLEAN := FALSE
     ,p_ignore_shortages        IN       BOOLEAN := FALSE);


/*================================================================================
==
    Procedure
      uncertify_batch
    Description
      This procedure reverts a completed batch to WIP and all the phantom
batches.

    Parameters
      p_batch_header (R)        The batch header row to identify the batch
                                Following columns are used from this row.
                                batch_id  (R)

      x_batch_header            The batch header that is returned, with all the
data
      x_return_status           outcome of the API call
                                S - Success
                                E - Error
                                U - Unexpected Error

================================================================================
*/

   PROCEDURE uncertify_batch (
      p_api_version        IN       NUMBER := gme_api_pub.api_version
     ,p_validation_level   IN       NUMBER := gme_api_pub.max_errors
     ,p_init_msg_list      IN       BOOLEAN := FALSE
     ,p_commit             IN       BOOLEAN := FALSE
     ,x_message_count      OUT      NUMBER
     ,x_message_list       OUT      VARCHAR2
     ,x_return_status      OUT      VARCHAR2
     ,p_batch_header       IN       gme_batch_header%ROWTYPE
     ,x_batch_header       OUT      gme_batch_header%ROWTYPE);


/*================================================================================
==
    Procedure
      uncertify_step
    Description
      This procedure reverts a step to WIP.

    Parameters
      p_batch_step (R)          The batch step row to identify the step
                                Following columns are used from this row.
                                batchstep_id  (R)
                                actual_start_date (O) (In case of direct
completion)
                                actual_cmplt_date (O)

      x_batch_step              The batch step that is returned, with all the
data
      x_return_status           outcome of the API call
                                S - Success
                                E - Error
                                U - Unexpected Error

================================================================================
*/

   PROCEDURE uncertify_step (
      p_api_version        IN       NUMBER := gme_api_pub.api_version
     ,p_validation_level   IN       NUMBER := gme_api_pub.max_errors
     ,p_init_msg_list      IN       BOOLEAN := FALSE
     ,p_commit             IN       BOOLEAN := FALSE
     ,x_message_count      OUT      NUMBER
     ,x_message_list       OUT      VARCHAR2
     ,x_return_status      OUT      VARCHAR2
     ,p_batch_step         IN       gme_batch_steps%ROWTYPE
     ,x_batch_step         OUT      gme_batch_steps%ROWTYPE);


/*==============================================================================
==
    Procedure
      close_batch
    Description
      This procedure closes batch and all the phantom batches.
       It also closes all the steps.

    Parameters
      p_batch_header (R)        The batch header row to identify the batch
                                Following columns are used from this row.
                                batch_id  (R)
                                batch_close_date (O)

      x_batch_header            The batch header that is returned, with all the
data
      x_return_status           outcome of the API call
                                S - Success
                                E - Error
                                U - Unexpected Error

================================================================================
*/

   PROCEDURE close_batch (
      p_api_version        IN       NUMBER := gme_api_pub.api_version
     ,p_validation_level   IN       NUMBER := gme_api_pub.max_errors
     ,p_init_msg_list      IN       BOOLEAN := FALSE
     ,p_commit             IN       BOOLEAN := FALSE
     ,x_message_count      OUT      NUMBER
     ,x_message_list       OUT      VARCHAR2
     ,x_return_status      OUT      VARCHAR2
     ,p_batch_header       IN       gme_batch_header%ROWTYPE
     ,x_batch_header       OUT      gme_batch_header%ROWTYPE);


/*==============================================================================
==
    Procedure
      close_step
    Description
      This procedure closes step.

    Parameters
      p_batch_step (R)          The batch step row to identify the step
                                Following columns are used from this row.
                                batchstep_id  (R)
                                step_close_date (O)
      p_delete_pending (R)      Delete the pending allocations if any for the
                                material lines associated with the step.

      x_batch_step              The batch step that is returned, with all the
data
      x_return_status           outcome of the API call
                                S - Success
                                E - Error
                                U - Unexpected Error
```

```
================================================================================
*/

   PROCEDURE close_step (
      p_api_version        IN       NUMBER := gme_api_pub.api_version
     ,p_validation_level   IN       NUMBER := gme_api_pub.max_errors
     ,p_init_msg_list      IN       BOOLEAN := FALSE
     ,p_commit             IN       BOOLEAN := FALSE
     ,x_message_count      OUT      NUMBER
     ,x_message_list       OUT      VARCHAR2
     ,x_return_status      OUT      VARCHAR2
     ,p_batch_step         IN       gme_batch_steps%ROWTYPE
     ,p_delete_pending   IN    BOOLEAN := FALSE
     ,x_batch_step         OUT      gme_batch_steps%ROWTYPE);


/*==============================================================================
==
    Procedure
      cancel_batch
    Description
      This procedure cancels batch and all the phantom batches.
       It also cancels all the steps.

    Parameters
      p_batch_header (R)        The batch header row to identify the batch
                                Following columns are used from this row.
                                batch_id  (R)

      x_batch_header            The batch header that is returned, with all the
data
      x_return_status           outcome of the API call
                                S - Success
                                E - Error
                                U - Unexpected Error

================================================================================
*/

   PROCEDURE cancel_batch (
      p_api_version        IN       NUMBER := gme_api_pub.api_version
     ,p_validation_level   IN       NUMBER := gme_api_pub.max_errors
     ,p_init_msg_list      IN       BOOLEAN := FALSE
     ,p_commit             IN       BOOLEAN := FALSE
     ,x_message_count      OUT      NUMBER
     ,x_message_list       OUT      VARCHAR2
     ,x_return_status      OUT      VARCHAR2
     ,p_batch_header       IN       gme_batch_header%ROWTYPE
     ,x_batch_header       OUT      gme_batch_header%ROWTYPE);


/*================================================================================
==
    Procedure
      reopen_batch
    Description
      This procedure reopens batch and all the phantom batches.
       It also reopens all the steps, if requested so.

    Parameters
      p_batch_header (R)        The batch header row to identify the batch
                                Following columns are used from this row.
                                batch_id  (R)
      p_reopen_steps (O)        Reopen all the steps.

      x_batch_header            The batch header that is returned, with all the
data
      x_return_status           outcome of the API call
                                S - Success
                                E - Error
                                U - Unexpected Error

================================================================================
*/

   PROCEDURE reopen_batch (
      p_api_version        IN       NUMBER := gme_api_pub.api_version
     ,p_validation_level   IN       NUMBER := gme_api_pub.max_errors
     ,p_init_msg_list      IN       BOOLEAN := FALSE
     ,p_commit             IN       BOOLEAN := FALSE
     ,x_message_count      OUT      NUMBER
     ,x_message_list       OUT      VARCHAR2
     ,x_return_status      OUT      VARCHAR2
     ,p_batch_header       IN       gme_batch_header%ROWTYPE
     ,p_reopen_steps       IN       BOOLEAN := FALSE
     ,x_batch_header       OUT      gme_batch_header%ROWTYPE);


/*================================================================================
==
    Procedure
      reopen_step
    Description
      This procedure reopens step.

    Parameters
      p_batch_step (R)          The batch header row to identify the batch
                                Following columns are used from this row.
                                batchstep_id  (R)

      x_batch_header            The batch header that is returned, with all the
data
      x_return_status           outcome of the API call
                                S - Success
                                E - Error
                                U - Unexpected Error

================================================================================
*/

   PROCEDURE reopen_step (
      p_api_version        IN       NUMBER := gme_api_pub.api_version
     ,p_validation_level   IN       NUMBER := gme_api_pub.max_errors
     ,p_init_msg_list      IN       BOOLEAN := FALSE
     ,p_commit             IN       BOOLEAN := FALSE
     ,x_message_count      OUT      NUMBER
     ,x_message_list       OUT      VARCHAR2
     ,x_return_status      OUT      VARCHAR2
     ,p_batch_step         IN       gme_batch_steps%ROWTYPE
     ,x_batch_step         OUT      gme_batch_steps%ROWTYPE);


/*==============================================================================
==
    Procedure
      reroute_batch
    Description
      This procedure reroutes batch (typically change the route associated with
the batch).

    Parameters
      p_batch_header (R)        The batch header row to identify the batch
                                Following columns are used from this row.
                                batch_id  (R)
      p_validity_rule_id (R)    Recipe validity rule id for the new recipe.

      x_batch_header            The batch header that is returned, with all the
data
      x_return_status           outcome of the API call
                                S - Success
                                E - Error
                                U - Unexpected Error

================================================================================
*/

   PROCEDURE reroute_batch (
      p_api_version        IN       NUMBER := gme_api_pub.api_version
     ,p_validation_level   IN       NUMBER := gme_api_pub.max_errors
     ,p_init_msg_list      IN       BOOLEAN := FALSE
     ,p_commit             IN       BOOLEAN := FALSE
     ,x_message_count      OUT      NUMBER
     ,x_message_list       OUT      VARCHAR2
     ,x_return_status      OUT      VARCHAR2
     ,p_batch_header       IN       gme_batch_header%ROWTYPE
     ,p_validity_rule_id   IN       NUMBER
     ,x_batch_header       OUT      gme_batch_header%ROWTYPE);


/*==============================================================================
==
    Procedure
      reschedule_batch
    Description
      This procedure reschedules batch and all the phantom batches.
       It also reschedules all the steps, if requested so.

    Parameters
      p_batch_header (R)        The batch header row to identify the batch
                                Following columns are used from this row.
                                batch_id  (R)
                                plan_start_date (R)
                                plan_cmplt_date (R)

      x_batch_header            The batch header that is returned, with all the
data
      x_return_status           outcome of the API call
                                S - Success
                                E - Error
                                U - Unexpected Error

=================================================================================
*/

   PROCEDURE reschedule_batch (
      p_api_version        IN       NUMBER := gme_api_pub.api_version
     ,p_validation_level   IN       NUMBER := gme_api_pub.max_errors
     ,p_init_msg_list      IN       BOOLEAN := FALSE
     ,p_commit             IN       BOOLEAN := FALSE
     ,x_message_count      OUT      NUMBER
     ,x_message_list       OUT      VARCHAR2
     ,x_return_status      OUT      VARCHAR2
     ,p_batch_header       IN       gme_batch_header%ROWTYPE
     ,x_batch_header       OUT      gme_batch_header%ROWTYPE);


/*===============================================================================
==
    Procedure
      reschedule_step
    Description
      This procedure reschedules step and all the subsequent steps, if requested
so.

    Parameters
      p_batch_step (R)          The batch step row to identify the step
                                Following columns are used from this row.
                                batchstep_id  (R)
                                plan_start_date (R)
                                plan_cmplt_date (R)
      p_reschedule_other (O)    Whether to reschedule subsequent steps.
      x_batch_step              The batch step that is returned, with all the
data
      x_return_status           outcome of the API call
                                S - Success
                                E - Error
                                U - Unexpected Error

=================================================================================
*/

   PROCEDURE reschedule_step (
      p_api_version        IN       NUMBER := gme_api_pub.api_version
     ,p_validation_level   IN       NUMBER := gme_api_pub.max_errors
     ,p_init_msg_list      IN       BOOLEAN := FALSE
     ,p_commit             IN       BOOLEAN := FALSE
     ,x_message_count      OUT      NUMBER
     ,x_message_list       OUT      VARCHAR2
     ,x_return_status      OUT      VARCHAR2
     ,p_batch_step         IN       gme_batch_steps%ROWTYPE
     ,p_reschedule_other   IN       BOOLEAN := TRUE
     ,x_batch_step         OUT      gme_batch_steps%ROWTYPE);


/*================================================================================
==
    Procedure
      scale_batch
    Description
      This procedure scaless batch up or down and all the phantom batches.

    Parameters
      p_batch_header (R)        The batch header row to identify the batch
                                Following columns are used from this row.
                                batch_id  (R)
      p_scale_factor (R)        How much to scale. (scale multiplier;
                                to make the twice as much quantity,
                                scale factor = 2; to reduce quantity to
                                half scale factor = -0.5.
      p_primaries (R)           Scaling based on product or Ingredients
                                INPUT  - Ingredients
                                OUTPUT - Products
      p_qty_type (O)            Whether to use formula quantities or batch
quantities
                                0 - Formula
                                1 - Batch   (Default)

      x_batch_header            The batch header that is returned, with all the
data
      x_over_allocations        Tables of material lines, where trying to scale
down
                                 the batch, and the quantities are going below
allocations
      x_return_status           outcome of the API call
                                S - Success
                                E - Error
                                U - Unexpected Error
                                G - Over allocation exists
```

```
================================================================================
*/

   PROCEDURE scale_batch (
      p_api_version        IN       NUMBER := gme_api_pub.api_version
     ,p_validation_level   IN       NUMBER := gme_api_pub.max_errors
     ,p_init_msg_list      IN       BOOLEAN := FALSE
     ,p_commit             IN       BOOLEAN := FALSE
     ,x_message_count      OUT      NUMBER
     ,x_message_list       OUT      VARCHAR2
     ,x_return_status      OUT      VARCHAR2
     ,x_over_allocations   OUT      gme_api_pub.unallocated_materials_tab
     ,p_batch_header       IN       gme_batch_header%ROWTYPE
     ,x_batch_header       OUT      gme_batch_header%ROWTYPE
     ,p_scale_factor       IN       NUMBER
     ,p_primaries          IN       VARCHAR2
     ,p_qty_type           IN       NUMBER DEFAULT 1);


/*==============================================================================
==
    Procedure
      theoretical_yield_batch
    Description
      This procedure calculates theoretical yield for the batch, and updates the
       quantities for the product lines.

    Parameters
      p_batch_header (R)        The batch header row to identify the batch
                                Following columns are used from this row.
                                batch_id  (R)
      p_scale_factor (R)        Theoretical yield in fractions

      x_batch_header            The batch header that is returned, with all the
data
      x_return_status           outcome of the API call
                                S - Success
                                E - Error
                                U - Unexpected Error

================================================================================
*/

   PROCEDURE theoretical_yield_batch (
      p_api_version        IN       NUMBER := gme_api_pub.api_version
     ,p_validation_level   IN       NUMBER := gme_api_pub.max_errors
     ,p_init_msg_list      IN       BOOLEAN := FALSE
     ,x_message_count      OUT      NUMBER
     ,x_message_list       OUT      VARCHAR2
     ,p_commit             IN       BOOLEAN := FALSE
     ,x_return_status      OUT      VARCHAR2
     ,p_batch_header       IN       gme_batch_header%ROWTYPE
     ,p_scale_factor       IN       NUMBER);


/*================================================================================
==
    Procedure
      allocate_batch
    Description
      This procedure auto-allocates batch and all the phantom batches.

    Parameters
      p_batch_header (R)        The batch header row to identify the batch
                                Following columns are used from this row.
                                batch_id  (R)
      p_alloc_type (O)          Whether to allocate lines with the
auto-allocation
                                 or user-initiated allocations class
                                0 - User initiated allocation class
                                1 - Auto allocation class
                                9 - All of the above  (Default)
      p_release_type (O)        Whether to allocate lines with consumption type
                                 of auto, manual etc.
                                0 - Automatic
                                1 - Manual
                                2 - Incremental
                                3 - by Step
                                9 - All of the above (Default)

      p_del_exist_alloc (O)     Delete existing allocations before, auto
allocationg.
                                True
                                False (Default)

      x_batch_header            The batch header that is returned, with all the
data
      x_unallocated_material    Table of materials, if some of the lines did not
get
                                 allocated.
      x_return_status           outcome of the API call
                                S - Success
                                E - Error
                                U - Unexpected Error
                                N - Items failed auto allocation

================================================================================
*/

   PROCEDURE allocate_batch (
      p_api_version            IN       NUMBER := gme_api_pub.api_version
     ,p_validation_level       IN       NUMBER := gme_api_pub.max_errors
     ,p_init_msg_list          IN       BOOLEAN := FALSE
     ,p_commit                 IN       BOOLEAN := FALSE
     ,x_message_count          OUT      NUMBER
     ,x_message_list           OUT      VARCHAR2
     ,x_return_status          OUT      VARCHAR2
     ,p_batch_header           IN       gme_batch_header%ROWTYPE
     ,x_unallocated_material   OUT      gme_api_pub.unallocated_materials_tab
     ,p_release_type           IN       NUMBER DEFAULT 9
     ,p_alloc_type             IN       NUMBER DEFAULT 9
     ,p_del_exist_alloc        IN       BOOLEAN DEFAULT FALSE);


/*================================================================================
==
    Procedure
      allocate_line
    Description
      This procedure auto-allocates material detail line.

    Parameters
      p_material_details (R)    The material detail row to identify the material
                                Following columns are used from this row.
                                material_detail_id  (R)
      p_alloc_type (O)          Whether to allocate lines with the
auto-allocation

                                 or user-initiated allocations class
                                0 - User initiated allocation class
                                1 - Auto allocation class
                                9 - All of the above (Default)
      p_del_exist_alloc (O)     Delete existing allocations before, auto
allocationg.

                                True
                                False (Default)
      x_unallocated_material    Table of materials, if some of the lines did not
get
                                 allocated.
      x_return_status           outcome of the API call
                                S - Success
                                E - Error
                                U - Unexpected Error
                                N - Items failed auto allocation

================================================================================
*/

   PROCEDURE allocate_line (
      p_material_details   IN       gme_material_details%ROWTYPE
     ,p_alloc_type         IN       NUMBER DEFAULT 9
     ,p_api_version        IN       NUMBER := gme_api_pub.api_version
     ,p_validation_level   IN       NUMBER := gme_api_pub.max_errors
     ,p_init_msg_list      IN       BOOLEAN := FALSE
     ,p_commit             IN       BOOLEAN := FALSE
     ,x_message_count      OUT      NUMBER
     ,x_message_list       OUT      VARCHAR2
     ,x_return_status      OUT      VARCHAR2
     ,p_del_exist_alloc    IN       BOOLEAN := FALSE);


/*==============================================================================
==
    Procedure
      convert_fpo
    Description
      This procedure is used to convert the firm planned order to one of many
batches.

    Parameters
      p_batch_header (R)        The FPO row to identify the FPO
                                Following columns are used from this row.
                                batch_id  (R)
      p_batch_size (R)          The size of the batch to be created.
      p_batch_size_uom (R)      The unit of measure in which the batch size is
passed in.
      p_num_batches (R)         Number of batches to be created.
      p_validity_rule_id (R)    The validity rule to be used to create the batch
(if
                               converting to 1 batch or p_use_for_all is set to
```

```
True)
      p_validity_rule_tab (O)   The validity rules tab (if converting to
multiple batches)
      p_leadtime (O)            The batch duration in case routing data or
                                 production rules does not exists.
      p_batch_offset (O)        The offset time between batches.
      p_offset_type (O)         The offset type.
                                0 - Start to start
                                1 - Finish to start
      p_schedule_method (O)     'FORWARD' or 'BACKWARD'
      p_plan_start_date (R)     The start date of the first batch
      p_plan_cmplt_date (O)     The completion date of the last batch
      p_use_for_all     (O)     Use the same validity rule for all the batches.

      x_return_status           outcome of the API call
                                S - Success
                                E - Error
                                U - Unexpected Error
                                N - Items failed auto allocation
                                V - Inventory shortages exist

================================================================================
*/

   PROCEDURE convert_fpo (
      p_api_version          IN       NUMBER := gme_api_pub.api_version
     ,p_validation_level     IN       NUMBER := gme_api_pub.max_errors
     ,p_init_msg_list        IN       BOOLEAN := FALSE
     ,p_commit               IN       BOOLEAN := FALSE
     ,x_message_count        OUT      NUMBER
     ,x_message_list         OUT      VARCHAR2
     ,p_enforce_vldt_check   IN       BOOLEAN
     ,x_return_status        OUT      VARCHAR2
     ,p_batch_header         IN       gme_batch_header%ROWTYPE
     ,x_batch_header         OUT      gme_batch_header%ROWTYPE
     ,p_batch_size           IN       NUMBER
     ,p_batch_size_uom       IN       VARCHAR2
     ,p_num_batches          IN       NUMBER
     ,p_validity_rule_id     IN       NUMBER
     ,p_validity_rule_tab    IN       gme_api_pub.recipe_validity_rule_tab
     ,p_leadtime             IN       NUMBER
     ,p_batch_offset         IN       NUMBER
     ,p_offset_type          IN       NUMBER
     ,p_schedule_method      IN       VARCHAR2
     ,p_plan_start_date      IN       gme_batch_header.plan_start_date%TYPE
     ,p_plan_cmplt_date      IN       gme_batch_header.plan_cmplt_date%TYPE
     ,p_use_for_all          IN       BOOLEAN := TRUE);


/*================================================================================
==
    Procedure
      delete_step
    Description
      This procedure deletes the step associated with the batch

    Parameters
      p_batch_step (R)          The batch step row to identify the step
                                Following columns are used from this row.
                                batchstep_id  (R)
      x_return_status           outcome of the API call
                                S - Success
                                E - Error
                                U - Unexpected Error

=================================================================================
*/

   PROCEDURE delete_step (
      p_api_version        IN       NUMBER := gme_api_pub.api_version
     ,p_validation_level   IN       NUMBER := gme_api_pub.max_errors
     ,p_init_msg_list      IN       BOOLEAN := FALSE
     ,p_commit             IN       BOOLEAN := FALSE
     ,x_message_count      OUT      NUMBER
     ,x_message_list       OUT      VARCHAR2
     ,x_return_status      OUT      VARCHAR2
     ,p_batch_step         IN       gme_batch_steps%ROWTYPE);


/*================================================================================
==
    Procedure
      insert_step
    Description
      This procedure inserts the new step to the batch

    Parameters
      p_batch_header (O)        The batch row to identify batch.
      p_batch_step (R)          The batch step row to insert to the batch.
      x_batch_step              The batch step that is returned, with all the
```

```
data
      x_return_status           outcome of the API call
                                S - Success
                                E - Error
                                U - Unexpected Error

================================================================================
*/

   PROCEDURE insert_step (
      p_api_version        IN       NUMBER := gme_api_pub.api_version
     ,p_validation_level   IN       NUMBER := gme_api_pub.max_errors
     ,p_init_msg_list      IN       BOOLEAN := FALSE
     ,p_commit             IN       BOOLEAN := FALSE
     ,x_message_count      OUT      NUMBER
     ,x_message_list       OUT      VARCHAR2
     ,x_return_status      OUT      VARCHAR2
     ,p_batch_header       IN       gme_batch_header%ROWTYPE
     ,p_batch_step         IN       gme_batch_steps%ROWTYPE
     ,x_batch_step         OUT      gme_batch_steps%ROWTYPE);


/*================================================================================
==
    Procedure
      partial_cert_batch
    Description
      This procedure is used to incrementally backflush the qty to the material
line.

    Parameters
      p_batch_header (R)        The batch header row to identify the batch
                                Following columns are used from this row.
                                batch_id  (R)
      p_material_details (R)    The material detail row to identify the material
line.
                                Following columns are used from this row.
                                material_detail_id  (R)
      p_qty (R)                 The quantity to apply incrementally.
      p_qty_type (R)            0 - By increment qty
                                1 - New actual qty
                                2 - % of Plan
      p_backflush_phantoms (R)  Backflush the quantities to the phantoms
associated
                                with the batch.
      p_ignore_shortages (R)    Ignore any inventory shortages.
      p_adjust_cmplt (R)        Adjust completed batches.

      x_unallocated_material    Table of materials, if some of the lines did not
get
                                 allocated.
      x_return_status           outcome of the API call
                                S - Success
                                E - Error
                                U - Unexpected Error
                                V - Inventory shortages
                                N - Items failed auto allocation

================================================================================
*/

   PROCEDURE partial_cert_batch (
      p_api_version            IN       NUMBER := gme_api_pub.api_version
     ,p_validation_level       IN       NUMBER := gme_api_pub.max_errors
     ,p_init_msg_list          IN       BOOLEAN := FALSE
     ,p_commit                 IN       BOOLEAN := FALSE
     ,x_message_count          OUT      NUMBER
     ,x_message_list           OUT      VARCHAR2
     ,x_return_status          OUT      VARCHAR2
     ,p_batch_header           IN       gme_batch_header%ROWTYPE
     ,p_material_details       IN       gme_material_details%ROWTYPE
     ,p_qty                    IN       NUMBER
     ,p_qty_type               IN       NUMBER
     ,p_backflush_phantoms     IN       BOOLEAN DEFAULT FALSE
     ,p_ignore_shortages       IN       BOOLEAN DEFAULT FALSE
     ,p_adjust_cmplt           IN       BOOLEAN DEFAULT TRUE
     ,x_unallocated_material   OUT      gme_api_pub.unallocated_materials_tab);


/*==============================================================================
==
    Procedure
      insert_material_line
    Description
      This procedure is used to insert the material line to the batch.

    Parameters
      p_material_detail (R)     The material detail row to insert the material
line.
      p_batchstep_no (O)        The batch step to which the material line should
```

```
be associated.
      x_material_detail         The material detail row with all the data
assigned.
      x_return_status           outcome of the API call
                                S - Success
                                E - Error
                                U - Unexpected Error

================================================================================
*/

   PROCEDURE insert_material_line (
      p_api_version        IN       NUMBER := gme_api_pub.api_version
     ,p_validation_level   IN       NUMBER := gme_api_pub.max_errors
     ,p_init_msg_list      IN       BOOLEAN := FALSE
     ,p_commit             IN       BOOLEAN := FALSE
     ,x_message_count      OUT      NUMBER
     ,x_message_list       OUT      VARCHAR2
     ,x_return_status      OUT      VARCHAR2
     ,p_material_detail    IN       gme_material_details%ROWTYPE
     ,p_batchstep_no       IN       NUMBER DEFAULT NULL
     ,x_material_detail    OUT      gme_material_details%ROWTYPE);


/*==============================================================================
==
    Procedure
      update_material_line
    Description
      This procedure is used to update the material line in the batch.

    Parameters
      p_material_detail (R)     The material detail row with the values to
update.
      p_values_tab (R)          The columns in the material line which should be
updated.
      p_scale_phantom (O)
      x_material_detail         The material detail row with all the data
assigned.
      x_return_status           outcome of the API call
                                S - Success
                                E - Error
                                U - Unexpected Error

================================================================================
*/

   PROCEDURE update_material_line (
      p_api_version        IN       NUMBER := gme_api_pub.api_version
     ,p_validation_level   IN       NUMBER := gme_api_pub.max_errors
     ,p_init_msg_list      IN       BOOLEAN := FALSE
     ,p_commit             IN       BOOLEAN := FALSE
     ,x_message_count      OUT      NUMBER
     ,x_message_list       OUT      VARCHAR2
     ,x_return_status      OUT      VARCHAR2
     ,p_material_detail    IN       gme_material_details%ROWTYPE
     ,p_values_tab         IN       field_values_tab
     ,p_scale_phantom      IN       BOOLEAN := FALSE
     ,x_material_detail    OUT      gme_material_details%ROWTYPE);


/*================================================================================
==
    Procedure
      delete_material_line
    Description
      This procedure is used to delete the material line in the batch.

    Parameters
      p_material_detail (R)     The material detail row to be deleted
      x_return_status           outcome of the API call
                                S - Success
                                E - Error
                                U - Unexpected Error

================================================================================
*/

   PROCEDURE delete_material_line (
      p_api_version        IN       NUMBER := gme_api_pub.api_version
     ,p_validation_level   IN       NUMBER := gme_api_pub.max_errors
     ,p_init_msg_list      IN       BOOLEAN := FALSE
     ,p_commit             IN       BOOLEAN := FALSE
     ,x_message_count      OUT      NUMBER
     ,x_message_list       OUT      VARCHAR2
     ,x_return_status      OUT      VARCHAR2
     ,p_material_detail    IN       gme_material_details%ROWTYPE);


/*================================================================================
==
    Procedure
      insert_line_allocation
    Description
      This procedure is used to insert a pending or complete line allocation for
an
      existing batch material line.

    Parameters
      p_tran_row (R)            The transaction row to be inserted.
      p_lot_no (O)       The lot no for the transaction
                                (Required if the item is lot controlled and lot
id
                                 is not passed in)
      p_sublot_no (O)       The sublot no for the transaction
                                (Required if the item is lot controlled and lot
id
                                 is not passed in)
      p_create_lot (O)TRUE, if the lot has to be created on fly.
      p_ignore_shortage (O)     TRUE, if any shortages have to be ignored.
      p_scale_phantom  (O)      TRUE, if the allocation is for phantom and it
has
                                to be scaled.
      x_material_detail         The updated material line.
      x_tran_row                The updated transaction row.
      x_def_tran_row            The default transaction row with any
adjustments.
      x_return_status           outcome of the API call
                                S - Success
                                E - Error
                                U - Unexpected Error

================================================================================
*/

   PROCEDURE insert_line_allocation (
      p_api_version        IN   NUMBER := gme_api_pub.api_version
     ,p_validation_level   IN   NUMBER := gme_api_pub.max_errors
     ,p_init_msg_list      IN   BOOLEAN := FALSE
     ,p_commit             IN   BOOLEAN := FALSE
     ,p_tran_row           IN   gme_inventory_txns_gtmp%ROWTYPE
     ,p_lot_no      INVARCHAR2 DEFAULT NULL
     ,p_sublot_no      INVARCHAR2 DEFAULT NULL
     ,p_create_lot   IN   BOOLEAN DEFAULT FALSE
     ,p_ignore_shortage   IN   BOOLEAN DEFAULT FALSE
     ,p_scale_phantom   IN   BOOLEAN DEFAULT FALSE
     ,x_material_detail   OUT  gme_material_details%ROWTYPE
     ,x_tran_row   OUT  gme_inventory_txns_gtmp%ROWTYPE
     ,x_def_tran_row   OUT  gme_inventory_txns_gtmp%ROWTYPE
     ,x_message_count      OUT  NUMBER
     ,x_message_list       OUT  VARCHAR2
     ,x_return_status   OUT  VARCHAR2);

END gme_api_pub;
/
COMMIT ;
EXIT;
```

Glossary

Application Programming Interface (API)

A documented, supported method for communicating within or between modules.

Business Object

An independent item of significance in the business world. An example of a business object is a sales order.

Business Process API

An API that performs a transaction for the calling module, e.g., to hire an employee, enter an order, or cost a material movement transaction.

Entity

An item of significance in the business world, that has no meaning without reference to a business object. An example of an entity is a sales order header. A Sales Order Header is an entity of the business object sales order.

Group API

An API intended for use by other authorized Oracle Applications.

Module

A module is a collection of one or more business objects and the associated transactions. A module publishes APIs for other modules and accesses other modules through their published APIs. An example of a module is Oracle Inventory.

Public API

An API intended for use by all applications.

Private API

An API intended for use by the owning module only.

Index