Oracle Projects and Billing Using Setups and Transaction Import from Third Party Systems - A NSF Case Study

Anupam Jain ajain@nsf.org
NSF International

Robert Kalvaitis kalvatis@nsf.orfg
NSF International

With 20,000 projects a year, which are setup using templates with tasks, rates, billing and revenue extensions and several third party systems performing the tasks in each project, it was necessary for NSF International to create an automatic process to post transactions, distribute usage, misc., labor costs and generate draft revenue and invoices. Once invoices are approved, transaction types are used in interface to AR. This is a case study of how we do it successfully and share the lessons learned.

About NSF International

NSF International, The Public Health and Safety Company™, (www.nsf.org) a not-for-profit, non-governmental organization, is the world leader in standards development, product certification, education, and risk-management for public health and safety. For 60 years, NSF has been committed to public health, safety, and protection of the environment. While focusing on food, water, indoor air, and the environment, NSF develops national standards, provides learning opportunities through its Center for Public Health Education, and provides third-party conformity assessment services while representing the interests of all stakeholders. The primary stakeholder groups include industry, the regulatory community, and the public at large.

NSF is widely recognized for its scientific and technical expertise in the health and environmental sciences. Its professional staff includes engineers, chemists, toxicologists, and environmental health professionals with broad experience both in public and private organizations.

NSF has earned the Collaborating Center designations by the World Health Organization (WHO) for Food and Water Safety and Indoor Environment. Serving manufacturers operating in 80 countries, NSF was founded in 1944 and is headquartered in Ann Arbor, MI USA. The NSF Mark is recognized for its value in international trade around the world and is respected by regulatory agencies at the local, state, and federal levels.

Why Projects and Oracle Projects

NSF has been using projects since 1998. As operations grew there were challenges that NSF faced. There were various third party systems that are developed. These are to meet the specific needs for each business area. There are systems to track and manage audits, laboratary management system, formulations and product composition system and certification system. As work was done in each of these systems it needed to be posted back to the financial systems. Prior to Oracle Projects this was done through time batches which relied on manual processes. There also was also a delay of about a week on an average. As the operations grew to overcome this and other issues and to have a scalable system ready for international operations NSF implemented Oracle E-business suite.

Projects are used to meet all the different needs for a customer / facility is managed and tracked at one place within the company. Suppose a client comes to us. He needs a product certified and also needs the facility certified for some for particular standards. So then we need a projects that has the 1to 4 audits that many be needed annually for the standard certification and then all the testing and formulations work needed for product certification for the product that the customer has. Some of this work may be billed monthly, some as work happens or in other combinations specific per customer and the contract signed. So a project is setup to meet these specific needs. Once this is done and classified as annual then it can be rolled every year. This enables NSF to manage the whole process efficiently and smoothly.

Oracle Projects– Setup & Use Details

Oracle Projects is the underlying foundation of the operational processes at NSF. It uses the Oracle Project Billing component of the suite and went live in December 2006. NSF handles over 20,000 projects per year with varying timing for both revenue recognition and invoicing (billing). There other complexities too. NSF has over 25 different programs or business units that cater to different customers. Within each unit multiple service offerings for multiple standards and certification needs for global clientele. There are offices in different parts of the world and the regulatory and standards requirements can be different. Billing is in multiple currencies too. All of this makes setting up an Oracle more difficult. To overcome this NSF has set up elaborate templates that help create projects easily. There are templates for each program and within that program for specific purposes. There is Standard Operating Procedures (SOP) for each group detailing the template to use for each purpose. The templates help create

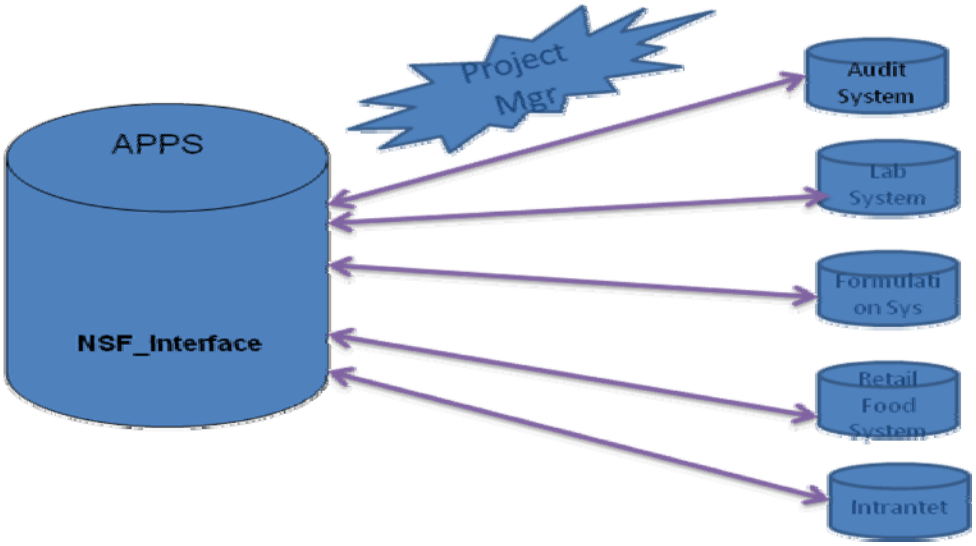
- Agreement
- Budget
- Extensions - both billing and revenue
- Tasks
- Expenditure type both labor and non labor with billable and non billable status
- Currency
- Rate schedules

Both Work/work and work/event projects are setup using templates. The implementation of Oracle has allowed NSF to greatly automate much of the project management, revenue, and billing processes.

Project information for Third Party Systems

After a project has been created and approved this project information needs to be made available for other NSF Systems. This is needed because the work is done in each of these systems based on the project. Whether it be audits, lab testing or formulation review. To achieve this NSF had done the following:

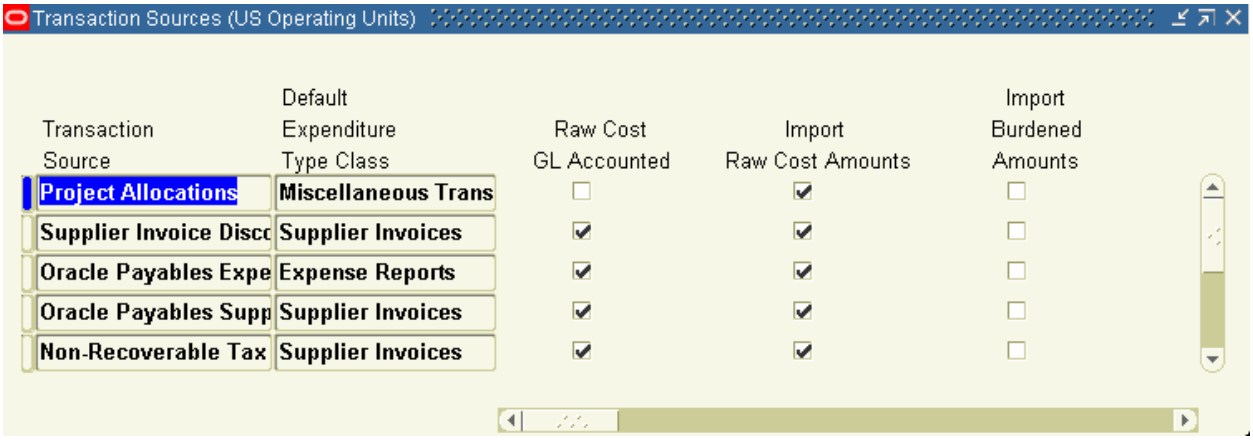
- Created a separate schema to keep custom database objects. This is called NSF_Interface.
- There are views created to feed this information to other systems. Please see addendum for the code of these view. That will also enumerate the oracle objects used.



Setup transaction Source

Transaction sources are set up in Oracle. One transaction source is setup for each system that NSF has. This identifies the transactions from each systems and helps manage the posting easily. This is setup in the projects responsibility

Setup →Expenditures →Transaction Sources



Import Transaction from 3rd party Systems

Since no work is truly done in E-business suite this work/transaction information needs to be gathered from the other systems. This is needed for Revenue recognition, billing and tracking. Custom code is written to get data from these applications and setup as concurrent programs, scheduled to run at night. These programs get the transaction information - edit/massage data and put it in the Oracle provided PA_Transaction_Interface_All table. The Oracle provided PRC Transaction import concurrent program is also scheduled soon after that to import these transactions into oracle. There are other process set to run after this. These include PRC distribute labor and usage cost, PRC generate draft revenue, PRC generate draft invoice for projects.

Due to these if the transaction is done in any system during the day then it gets recorded in Oracle the same night, revenue recognized and the invoice available for the 'key member' Project manager to approve the very next day.

Send Transaction Import Errors to Key members – Project Managers

After the transactions are imported into oracle the "PRC: Transaction Import" process marks the rejected transactions with an error status and a rejection code. NSF has mapped these rejection codes to a custom error message which are stored in a table. There is a custom concurrent program that loops through all rejected transaction and groups them by key member – Project manager. It will then send an e-mail to each key-member/Project manager with the list of error transaction and the error message. It also removes the error status so that it can be picked up for processing again that night. Project Managers can correct the errors and the transactions will be processed again overnight.

Conclusion

Being able to use the E-business suite and project foundation and interface it with other proprietary systems has given NSF the flexibility to manage its business with applications suited to each of its specific needs.

NSF is able to use the financial aspects of the system. It is been able to automate manual feeding of transaction from other systems. Revenue recognition and billing is much faster and also to have an invoice approval process from the project key members.

The accounting department and other back-office functions have gained substantial functionality and able to manage the customer and information, work status (WIP) more accurately cleanly and easily.

Addendum – Code listing for views that NSF created to send Project Information

```
CREATE OR REPLACE VIEW nsf_com_project_task_vw AS
     SELECT pa.project_id
            ,pa.segment1 project no
            ,pa.NAME project name
            ,pa.long_name project_long_name
            ,pa.start date project start date,
            ,pa.completion date project completion date,
            ,pa.closed_date project_closed_date, pt.task_id,
,pt.task number task no, pt.task name task name,
            ,pt.description task description,
            ,pt.service_type_code service_type_code
            ,pt.start date task start date,
            ,pt.completion date task_completion_date
            ,pc.customer_id
            ,pc.customer number customer no,
            ,pc.ship to address id ship to address id,
            ,NVL (csua3.LOCATION, csua.LOCATION) ship_location
            ,csua3.LOCATION task location
            ,pc.bill to address id bill to address_id
            ,csua2.LOCATION bill_to_location
            ,pcl.class code project category
            ,pcl2.class code program name, segment value program_code,
            ,UPPER (pps.project_status_name) project status
            ,GREATEST (pa.last update date,
                       pt.last update date,
                       pc.last_update_date,
                       csua.last update date,
                       csua2.last update date,
                       psvl.last_update_date,
                       psvls.last update date,
                       pcl.last update date,
                       pcl2.last_update_date
                      ) last update date
       FROM apps.pa project customers_v pc,
            apps.pa_projects_all pa,
            apps.pa tasks pt,
            apps.pa project classes pcl,
            apps.pa_project_classes pcl2,
            apps.pa segment value lookups psvl,
            apps.pa segment value lookup sets psvls,
            apps.hz_cust_site_uses_all csua,
            apps.hz cust site uses all csua2,
            apps.hz cust site uses all csua3,
            apps.pa_project_statuses pps
      WHERE pa.project id = pc.project id
        AND pa.project id = pt.project id
        AND pa.project_id = pcl.project_id
        AND pcl.class category = 'Project Category'
        AND pa.project id = pcl2.project id
        AND pcl2.class_category = 'Program'
      AND psvls.segment value lookup set id = psvl.segment value lookup set_id
        AND psvls.segment value lookup set name = 'Service Type to Program'
        AND psvl.segment_value_lookup = pt.service_type_code
        AND csua.cust acct site id = pc.ship_to_address_id
        AND csua.site use code = 'SHIP TO'
        AND csua2.cust_acct_site_id = pc.bill_to_address_id
        AND csua2.site use code = 'BILL TO'
        AND csua3.cust acct site id(+) = pt.address id
        AND NVL (pa.project_status_code, '@') != 'UNAPPROVED'
        AND NVL (pa.template flag, '@') = 'N'
        AND pa.project status code = pps.project_status_code
   ORDER BY project_no, task_no;
```

```
CREATE OR REPLACE FORCE VIEW nsf_com_project_task_expend_vw AS
   SELECT ptc.project id
          ,ptc.task id
          ,ptc.expenditure type
          ,pet.description expenditure_description
          ,pet.revenue category code
          ,SUBSTR(apps.nsf projects interface_pkg.get_exp_type_class
                     (ptc.expenditure_type),
                   1,
                   250
                  ) expenditure type class,
           ,MIN (ptc.start date active) exp start date
           ,MAX (ptc.end date active) exp_end_date
           ,ptc.chargeable flag
           ,ptc.billable indicator
           ,ptc.expenditure category,
           ,GREATEST (MAX (ptc.last update date),
                      MAX (pet.last update_date),
                      pt.last update date
                     ) last update date
       FROM apps.pa transaction controls ptc,
            apps.pa expenditure types pet,
            nsf com project task vw pt
      WHERE ptc.expenditure type = pet.expenditure_type
        AND ptc.expenditure type IS NOT NULL
        AND ptc.task id IS NOT NULL
        AND ptc.project id = pt.project_id
        AND ptc.task id = pt.task_id
   GROUP BY ptc.project id,
            ptc.chargeable flag,
            ptc.billable_indicator,
            ptc.task id,
            ptc.expenditure category,
            ptc.expenditure type,
            pet.description,
            pet.revenue category_code,
            pt.last_update_date;


CREATE OR REPLACE FORCE VIEW nsf interface.nsf proj task exp rates_vw AS
   SELECT pa.segment1 project number, pt.task_number task_number,
          ptc.expenditure_type,
          TO NUMBER
             (nsf utl pkg.get bill rate (ptc.expenditure_type,
                                         ptc.task id,
                                         pt.non lab std bill rt sch id,
                                         pt.non_labor_bill_rate_org_id
                                        )
             ) bill_rate,
          TO NUMBER
             (nsf utl pkg.get markup percentage
                                              (ptc.expenditure_type,
                                               ptc.task id,
                                               pt.non lab std bill rt sch id,
                                               pt.non_labor_bill_rate_org_id
                                              )
             ) markup_percentage,
          TO NUMBER
             (NVL
                 (NVL
                     (nsf utl pkg.get discount percentage
                                              (ptc.expenditure_type,
```

```
                                                ptc.task id,
                                                pt.non lab std bill rt sch id,
                                                pt.non_labor_bill_rate_org_id
                                               ),
                        pt.non_labor_schedule_discount
                       ),
                pa.non_labor_schedule_discount
               )
          ) discount percentage,
       pc.customer number customer no,
       NVL (csua3.LOCATION, csua.LOCATION) ship_location,
       csua2.LOCATION bill location,
       DECODE (SUBSTR (pt.task number, 1, 2),
               'CL', 'Chemistry Laboratory',
               'PL', 'Engineering Laboratory',
               'ML', 'Microbiology Laboratory',
               'TE', 'Toxicology',
               'PE', 'Engineering Laboratory',
               DECODE (SUBSTR (pt.task number, 1, 1),
                       'X', 'Chemistry Laboratory',
                       NULL
                      )
              ) department,
       ptc.billable indicator, papf.full name project manager,
       NVL (papf.employee number, papf.npw number) employee_number,
       UPPER (pps.project status name) project_status,
       segment value program code,
       ffvtl.description program name,
       pcl.class code project category,
       NVL(nsf_utl_pkg.get_rev_extension (ptc.project_id, ptc.task_id),
           'N'
          ) revenue extention,
       pa.start date project start date,
       pa.completion date project completion date,
       pa.closed date project closed date, pt.start_date task_start_date,
       pt.completion date task completion date,
       ptc.start date active exp start date,
       ptc.end date active exp end date
  FROM apps.pa project customers_v pc,
       apps.pa projects all pa,
       apps.pa tasks pt,
       apps.pa transaction controls ptc,
       apps.pa project classes pcl,
       apps.pa project classes pcl2  ,
       apps.pa segment value lookups psvl,
       apps.pa segment value lookup_sets psvls,
       apps.fnd flex values ffv,
       apps.fnd flex values tl ffvtl,
       apps.hz cust site uses all csua,
       apps.hz cust site uses all csua2,
       apps.hz cust site uses all csua3,
       apps.pa project statuses pps,
       apps.per all people f papf,
       apps.pa project players ppp
 WHERE  pa.project id = pc.project id
   AND pa.project id = pt.project_id
   AND pt.task id = ptc.task id
   AND pa.project id = pcl.project id
   AND pcl.class category = 'Project Category'
   AND pa.project id = pcl2.project id
   AND pcl2.class category = 'Program'
   AND psvls.segment value lookup set id = psvl.segment value lookup set_id
   AND psvls.segment value lookup set name = 'Service Type to Program'
   AND psvl.segment_value_lookup = pt.service_type_code
```

```
    AND ffvtl.flex value id = ffv.flex value_id
    AND ffv.flex value set id = 1010390
    AND ffv.enabled flag = 'Y'
    AND ffv.flex value = segment value
    AND csua.cust acct site id = pc.ship_to_address_id
    AND csua.site use code = 'SHIP TO'
    AND csua2.cust acct site id = pc.bill_to_address_id
    AND csua2.site use code = 'BILL TO'
    AND csua3.cust acct site id(+) = pt.address_id
    AND NVL (pa.template flag, '@') = 'N'
    AND pa.project status code = pps.project_status_code
    AND papf.person id = ppp.person id
    AND ppp.project role type = 'PROJECT MANAGER'
    AND ppp.project id = pa.project id
    AND NVL (ppp.start date active, SYSDATE) <= TRUNC (SYSDATE)
    AND NVL (ppp.end date active, SYSDATE) >= TRUNC (SYSDATE)
    AND NVL (papf.effective start date, SYSDATE) <= TRUNC (SYSDATE)
    AND NVL (papf.effective_end_date, SYSDATE) >= TRUNC (SYSDATE);
```

Addendum – Package specification to get transaction information from other systems.

```
CREATE OR REPLACE PACKAGE APPS.Nsf_Projects_Interface_Pkg AS

   FUNCTION get_exp_type_class(In_Exp_type IN VARCHAR2) RETURN VARCHAR2;

   FUNCTION exp_type_time(In_Exp_type IN VARCHAR2) RETURN BOOLEAN;

   PROCEDURE get pt trn data ( errbuf OUT NOCOPY VARCHAR2
                              ,retcode  OUT NOCOPY NUMBER);

   PROCEDURE get BKR trn data ( errbuf OUT NOCOPY VARCHAR2
                               ,retcode  OUT NOCOPY NUMBER);

   PROCEDURE get Oas trn data (errbuf OUT NOCOPY VARCHAR2
                              ,retcode  OUT NOCOPY NUMBER);

   PROCEDURE get org det( In application IN VARCHAR2
                        , In_project_no  IN VARCHAR2
                        , In task no  IN VARCHAR2
                        , v org id OUT NUMBER
                        , v_org_name OUT VARCHAR2);

   FUNCTION Revenue extn(In project no VARCHAR2,
                         In_task_no  VARCHAR2) RETURN BOOLEAN;

   FUNCTION Proj_completed (In_project_no VARCHAR2) RETURN BOOLEAN;

   PROCEDURE send tr import errors (errbuf OUT NOCOPY VARCHAR2
                                   ,retcode  OUT NOCOPY NUMBER
                                   ,in_Transaction_source VARCHAR2 DEFAULT
NULL);

END Nsf_Projects_Interface_Pkg;
```