

Implementation Ideas, Tips, and Techniques from TECT

Frank Boerger and Jim Crum

Turbine Engine Components Textron and BOSS Corporation

Introduction

Abstract: TECT implemented all Financial, Distribution, Manufacturing and HR/Payroll Applications at six sites, and this paper provides the new Oracle customer with implementation tips, techniques, and lessons learned. We will provide helpful information about project schedules, Oracle Support, the big bang approach, customizations, database administration, testing, training, and security.

About TECT and BOSS

TECT is a manufacturing division of Textron. We manufacture engine components for jet engines that are used in both civilian and high performance military aircraft. We use the Oracle Applications at the following sites: Georgia (2), Florida, Connecticut, California, and Ohio. We have about 175 users. The computing environment consists of R10.7SC16.1, two HP UNIX computers, three instances (2 test, 1 production). Installed applications are GL, AP, AR, PO, FA, OE, INV, BOM, ENG, WIP, MRP, CST, HR, and PAY. We also use multi-org.

BOSS Corporation is a services company that specializes in implementing and administering the Oracle Applications. The company was formed in 1995, has been growing rapidly, and currently has about 30 functional, technical, and database consultants. BOSS Corporation is the author of *Special Edition: Using Oracle Applications* published in April 2000 by Que (ISBN 0-7897-1280-6). This book is recommended reading if you would like additional implementation ideas, tips, and techniques.

Project Schedules

We implemented the manufacturing and financial applications at the first site in 11 months. Each of the mini-implementations at the other sites was a 3 or 4 month project. We used consulting resources for the first two sites and then implemented the remaining sites with our internal team.

The HRMS applications took longer than expected. We have 11 payrolls (6 salaried/bimonthly and 5 hourly/weekly). Since each plant has its own manufacturing management, each payroll implementation had its own dynamics and requirements for several policies like vacations, holidays, and absenteeism. We had to load mid year payroll balances, we had many interfaces to insurance and benefit providers, and we performed parallel testing until everything was perfect. We wrote several custom reports for payroll. In addition, we discovered retirees and survivors of retirees that had to be part of the implementation. Many of these retirees came from plants that had been closed or consolidated many years ago and there were several distinct pension plans.

Working with Oracle Support

Early in the project, we should have had an intro to Oracle database fundamentals class. We needed to understand the database concepts behind columns and tables. We needed to know what happens during a commit or a rollback. Midway through the initial implementation, we should have taken an introductory PL/SQL class. If we had understood these items, we would have done a better job of understanding and talking with support. We needed to be able to speak "Oracle-ese".

Your ability to work effectively with Oracle Support is a critical success factor for your implementation project. Working with Support is an acquired skill, and it is well worth your effort to develop this ability. Understanding what happened and in what order is important to diagnosing and solving an error. Document the error and be prepared to

lead the support analyst through exactly the same series of keystrokes to re-create the problem. Know the navigation path, the data that was entered, parameters and lists of values that were used, and so forth. Because an error might create several error messages, try to determine which event is the root cause because secondary messages might go away when you fix the basis of the problem.

In theory, the support process is supposed to work as follows. First, you contact Oracle World Wide Support and provide information for a technical assistance request (TAR). Next, the analyst gathers information and attempts an early statement and diagnosis of the problem. Third, the analyst reviews the existing knowledge base of previous problems with similar keywords. If the analyst cannot find references to similar TARs, you must decide to restate the problem, continue the diagnosis, or escalate to a more skilled analyst. Fourth, if the problem is identified in the knowledge base, the analyst determines whether a solution is available. If a patch is required, the analyst sends the patch to you. You apply the solution to a test system and determine whether the problem is resolved. Then, you close the TAR.

If the analyst can't identify the solution, you perform additional research and perhaps bring in additional resources. If progress becomes difficult, you must decide whether the business impact and experience level of the analyst justifies escalation of the problem. Finally, if you are certain you have correctly diagnosed the problem and a solution is not available, the analyst creates a bug record and forwards the problem to Oracle Development. At this point, you lose visibility of the problem resolution because neither you nor the analyst can affect Development. Development might respond with a patch or a determination that the release works as designed. If the program works as designed, the issue must be processed as an enhancement request.

Your job is to lead Oracle Support through this process, and if you follow the procedure, this process works pretty well. Often in the real world, however, the analyst first determines your version level of the executable causing the problem. He then recommends without further analysis that you upgrade to the latest certified revision for that executable.

This shortcut is often **NOT** in your best interest. This is the support version of "take two aspirin and call me in the morning." When you have a program change to the latest version, the procedure might involve multiple (dependent) patches or a Megapatch. This strategy represents a lot of work on your part and might not even make any sense. You get the latest version of a program, but you might patch and test for hours or days before you realize you didn't diagnose the problem and you must start all over again. This procedure can cause lots of stress in the case of an important business problem.

Implementation Methods

We employed two implementation techniques in our project. First, for the manufacturing and financial applications, we used an implementation strategy composed of one big bang implementation followed by five "cookie cutter" projects to implement the other manufacturing plants and our division headquarters. Second, for the HRMS and Payroll applications, we used an approach using six phases to implement the hourly and salaried payrolls at each of the 6 sites. Since our payroll is centralized and our manufacturing is decentralized, the work on HRMS applications was done at the division HQ and the first manufacturing and financials site was our Orlando manufacturing plant for 12 applications. The five cookie cutter projects for the manufacturing and financial applications followed the Orlando site. This technique turned out to be very efficient once we had worked out the basic configuration at the first site. We learned several lessons about the implementation technique:

The first thing we discovered was that we were tight on space for the implementation team. We separated the team into multiple rooms. That was our mistake. Enterprise Resource Planning (ERP) software implementation is a team sport. The Oracle Applications are integrated, and the implementation effort must be integrated too. We were uncoordinated for several months, but we really started making progress when we all crammed into one conference room.

Second, we didn't push each other hard enough. Maybe in retrospect, we see all of the things we could have done better, faster, and cheaper, but while you are involved in the day to day activities of the project, it is easy to lose momentum. Waiting on other team members to make a decision or Oracle support to fix a bug might be a good temporary excuse for why something didn't get done on any given day, but during an eleven month project, the time really adds up. You can eliminate that situation by producing detailed work plans for 2 to 3 weeks for the entire project team.

In addition, when you are starting a project, you don't know how long certain things will take. This software is based on technology, and sometimes it doesn't work right the first time. For example, we lengthened our first conference room pilot by almost eight weeks so we could do a thorough bug search in release 10.7 SmartClient. We couldn't leave our first site with bugs because of two potential problems. First, the bugs might force us to return to the first site when we were in the middle of the second implementation. That would destroy the schedule at the second site. Second, unresolved issues would give the software a bad reputation with the unimplemented sites, and that would make our cookie cutter implementations much more difficult. We had to take the time get it right on the first try.

Third, while the Orlando, Florida manufacturing plant was our first implementation site, division HQ was in Georgia. The project manager was also the Director of MIS for the division and he had responsibilities from his day job. When the project manager was at HQ, we would lose leadership momentum on the implementation project. Software implementation is a full time job, and it helps when the project manager can delegate work on the legacy system.

Next, we are pleased to report one of our critical success factors was support for the implementation by our division president. When we purchased the software, he offered us anyone in the division (including plant managers) to be on the implementation team. We assembled a team with 75 years of TECT business experience, and the team was cross-functional. These experienced employees were subject matter experts (SMEs) on our business requirements. In addition, our SMEs are really well connected to and trusted by management at all of the implementation sites. The selection of the SMEs was important, because everyone had to trust the Orlando site and the implementation team to get it right, so their implementation would be effective and efficient. The SMEs had to be able to think "outside of the box", and they needed to be as computer savvy as we could get.

Even though the secondary implementations were efficient copies of the original project, we think it is very important for each of the plants to staff their project teams with high level people. The project implementation team worked best with department heads, managers, and policy makers. We recommend you avoid the use of clerical team members, because the clerical staff can't take responsibility for the changes caused by an ERP project. Realistically, they probably do not possess the high-level vision necessary for success. Experienced department level managers will understand that the right decision might result in overall more work for their department, but that it supports the greater good.

Finally, we used the TECT implementation team members as filters between the users and the software. We didn't necessarily want the users to know or have to cope with every single function point of the Oracle Applications. This technique kept the complexity of the applications hidden with the project team and away from the users. This strategy allowed us to control the scope of the secondary implementations and avoid feature analysis paralysis.

Customization

Early in the project, we etched in stone that there would be no customizations (knowing full well there would be some). Customizations are defined as interfaces to non-Oracle systems, extensions to the applications, and modifications of Oracle software. By using the big bang approach we avoided all interfaces back to our legacy system (but we do have several interfaces to third party systems like bar code scanners and HRMS benefit providers).

We use Optio software to avoid customizing Oracle reports where we don't like the output. The Optio software looks like an operating system printer to the Oracle Applications, but it is installed between the operating system and the physical printer and allows us to reformat the Oracle output without touching the Oracle report program. We use

Optio on the payables check, purchase order, payroll check, payroll direct deposit advice, receivables invoice, shipping documents, and shop floor routing. To print the checks, we use blank check stock, MICR fonts, magnetic toner, and embedded signature graphics. To print the shop floor routing, we compressed a 50 page Oracle report into 5 pages, and we included bar codes for data collection at count points. We underestimated the amount of work on special forms, because we thought one PO, one paycheck, one receivables invoice, and one shop floor routing would serve all sites. In reality, as soon as we demonstrated that we could easily change the Optio output, each site wanted their version of the document. We created four, five, or six versions of each form, and we continue to refine these forms today.

We wrote some special reports especially for local requirements in the Payroll and Work in Process applications. We used experienced database programmers (without applications experience) that were less expensive than our functional applications consultants. These programmers wrote some complex code, and as transaction data built up in the database, the programs started running slower and slower. After about a year, we had technical consultants from our applications consulting firm tune several of the programs to keep them usable. This tuning process can take 2 to 4 days per program and was an unforeseen expense.

System Administration

We implemented a creative solution for our system administration requirements. We are located in a small town and there aren't a whole lot of Oracle and Unix gurus in our labor pool. Since our division has six locations in five states, we knew we would have a wide area network. Therefore, we put the hardware in a secure data center operated by EDS in Texas. This data center provides us with all of the requirements for the hardware and the Unix operating system. EDS operators perform daily back ups and monitor the system console for a wide variety of messages and system events.

We get database and applications administration services from BOSS Corporation, our applications services firm. We get access to a pool of four exceptional administrators, and we can escalate problems through their organization. Each day they dial in to our system and perform preventive maintenance. They have a large library of scripts and queries that locate and predict potential problems so they can be fixed before the users are affected. We think an excellent administrator can easily make a three to five percent improvement in system performance. When you leverage that improvement over our 175 users, it increases the performance of our users by about the equivalent of seven full time employees. When you consider that we would have difficulty hiring and retaining even one excellent DBA and we get access to a large pool of talent, our system administration is more stable and effective because we buy these services outside.

BOSS and EDS work well together. They have other clients in common, and there is a clear division of responsibility. Two or three times per year we have planning conference calls to support additional hardware or software changes.

Testing

We have three databases on two machines. Everything is tested in either the test or pilot databases before it is applied to the production database. The test and pilot databases are copies made from production about every three weeks. We don't do full regression testing on patches from Oracle Support, but we do test everything for reasonableness, and get a power user to sign off on the patch in a test database before it is applied to production.

We staged a comprehensive conference room pilot (CRP) of all integrated business processes before each site started using the production software. The pilot at the first site included a major search for release 10.7 bugs. We felt the CRP should serve two purposes. First, we wanted to validate that the software would work and meet the business requirements. Second, we wanted to test to see if we could make the system fail. We wanted to know if the problem was us, the configuration, or the software. The CRPs at the second through sixth sites were valuable for training and orienting the managers at each site.

Training and Communication

Education for our project team was a mix of on the job experience and classroom study. Each team member was paired with an experienced consultant and we learned by working through business scenarios with the consultant. We combined applications training for the TECT people with business training for the consultant, because the consultants needed to know our business to give us good advice. For example, I learned how Oracle does a bill of material while the consultant learned from me how our parts are engineered. We forced the training issue by setting scheduled times of day for working on each application in this workshop/laboratory approach.

Also, we often put the primary and secondary implementation team member for each application through Oracle education. For example, there was a primary team member for inventory, and I was the secondary for inventory. Since that was such a key module, we both went to the Oracle inventory class. We made one mistake with our education at the Oracle education centers. We should have had our experienced consultants give us a mini-course or at least a high-speed demonstration before we went to the Oracle class. We were in class as new release 10.7 customers. However, many classmates were experienced Oracle users, and they were preparing for upgrades from release 10.5 or 10.6. We felt left behind at times.

In addition, we didn't learn about the topical essays in the release 10 reference manuals until too late. These very valuable essays were mentioned at our project kickoff meeting, but we ignored the suggestion that we read several hundred pages of material to start the project. We should have read this material before attending classes at Oracle education. That would have given us a better context for the class material.

We have used the OAUG conferences from the beginning to help the implementation team. While these conferences were a bit overwhelming at first, and for some of us it was the first contact the Oracle applications, OAUG serves as part of the process of developing comfort levels between our Oracle champions and the software. This forum is extremely valuable. Make sure you stay connected with it. We have not missed a conference since we got involved with the applications.

Communication between the main implementation team and the management at each site was important. We had a 5 PM meeting every day to debrief the project team. We had an 8:30 AM organizing meeting with users when we were in pilot and transition mode. Everyone announced what he would be working on and what his accomplishments would be that day.

We started end user training with the assumption that everyone would absorb the changes and new technology at the same pace. That doesn't always happen. We had some remedial classes, and we had to make some hard choices on some people. User training was a mix of simple one-on-one instruction and learning by data entry.

The next to last site believed they had special business requirements, and they requested extra training before their project started. They wanted to make sure they could express their business requirements. They paid for three weeks of on site training and orientation from their own budget. The positive result of this training was that the site took responsibility very early in the implementation. As it developed, the Oracle software met their needs, but the extra training proved to be a very good idea. The last site took the extra training also.

Security

We have 6 organizations in one database instance and one set of books. We use multi-org to partition the AR, PO, AP, and OE data among the organizations. The customer list is short (after all how many jet engine manufacturers are there?) and customer service is centralized. Therefore, the division HQ owns the customers and each organization has its own customer sites. Since procurement is decentralized, we allow each site to create vendors, as they need them. Multi-org causes us to share the vendor records, but each organization has its own vendor sites.

We started with unrestricted menus for the consultants and limited menus for the TECT employees. Then we gave additional capability to each responsibility as it was needed and knowledge was gained. We implemented very

restricted menus for users. Requests for additional menus and responsibilities are still directed to the appropriate implementation team member, only after the end user, requesting manager, and data owner complete and co-sign a controlled form. We added data owner soon after this form was created as we were quick to realize that data integrity was paramount, and we needed all department level managers in the plants to help be responsible for keeping our hard earned database pristine.

Conclusion

In summary, we had a very successful Oracle applications implementation project at five manufacturing sites and one division headquarters. We solved a lot of problems and we are happy to share our experiences with you. Some of the things we think we did especially well include working well with Oracle Support, testing, using a variety of implementation methods, performing customization carefully, outsourcing our system administration, and training.

Our implementation team is now geographically disbursed, but it still functions as a team. That concept has sometimes been a difficult point to explain to our upper management who thought they would implement, then disband the team. Management has to be reminded that critical maintenance and support is an ongoing part of a complex system such as this. We have had two promotions and one retirement from our core group. We realize that the ERP software project did not end when the last plant went live with the production database. The work is a continuous process of improving, testing, patching, and upgrading. We are starting to plan our upgrade to release 11 or 11i while at this conference.

About the Authors

Frank Boerger is the MIS manager at Turbine Engine Components Textron. He has 25 years of experience in all facets of the TECT business. Frank was the lead team member for implementation of Oracle Engineering, Work in Process, and Order Entry.

Jim Crum is one of the founders and COO of BOSS Corporation. He is an active consultant for Oracle General Ledger, Receivables, Payables, Cost Management, and Multi-Org. Jim edits the tips and techniques newsletter for the Atlanta OAUG, and he is the lead author of the book *Special Edition: Using Oracle Applications* published in April 2000 by Que.