OBIEE Conference
Cal Poly San Luis Obispo
Lessons Learned

1. Should have come up with a catchy project name and icon.
2. Focus on delivering information via easy to use Web enabled dashboards rather than Web enabled query tools. There are a small percentage of users who want, need and will use the query tools but the great majority are happy with a dashboard.
3. In order to deliver good performance and work most effectively with OBIEE the Finance 9.0 delivered Finance Reporting Solution was converted into a set of Dimensional Modeled Data Marts.
4. Joint training with Functional Trainers/Analyst and Technical Developers was very beneficial and a great team builder.
5. Separate tools training from functional training
6. Early, Web based training for key individuals was worthwhile.
7. We should have established better technical contacts with Oracle earlier (if that was possible).
8. Traditional requirements gathering methodologies and processes don’t work as well for Interactive Dashboard development. A more interactive, joint application development approach using prototyping worked well for Finance.
9. The dashboard developers and data warehouse developers need to communicate regularly. The data warehouse programmers need to understand the goal of the dashboards and how the data will be represented. The dashboard developers need to understand how the data will be deployed to the data warehouse. Hierarchies, calculated values, grain of data, dates, etc. need to be understood by both.
10. Setting up a sandbox environment to allow the data warehouse team time to learn the tool was extremely valuable. The sandbox was set up fairly quickly while DEV was set up in a more structured manner.
11. Security is extremely complicated. The time it takes to understand all of the security components is extensive.
12. Utilizing the PeopleSoft roles to drive security is beneficial. We have automated the security within OBIEE. This method also allows the functional areas to manage user authorization by managing the users within PeopleSoft roles.
13. Deploying within a Linux environment may have caused issues within the environment (i.e., replication). It may be wiser to look at the Windows version.
14. Tying the implementation of dashboards to a tool replacement can be an issue. We are trying to move away from Brio Query. In some cases, we are being asked to duplicate 40-50 queries using the dashboard functionality for one area. This is proving to be difficult.
15. Moving from a query tool that allows flexibility in choosing data sources is problematic when moving to OBIEE. The metadata layer of OBIEE creates overhead for the data warehouse team. We are now looking to deploy BI Publisher to help with this issue.
16. Developing a method to manage the intake of enhancements/fixes is beneficial. Having a simple way to see the list, add items, prioritize items, know how many tasks have been completed, and how many outstanding items exist is important.
17. There is a lot of power in OBIEE. That power equates to complexity within the environment. Try to keep the first deployment simple while learning the tool and environment. You can always build from there.
18. Security by default allows everyone to see/do everything. Security needs to be set up to only allow authorized access.