Tim Bradley
Architecture
September 14, 2006

CONCURRENT ENGINEERING– SOFTWARE DESIGN

As the book points out, there are several important aspects of concurrent engineering. Requirements and specifications are the first artifacts to be generated. As the requirements are ironed out, architectural decisions are made. The system is broken down generally into hardware and software elements. At this point designing can take place concurrently: software can be designed and tested as hardware design takes place independently. Eventually, these concurrent paths are integrated together to unify the system.

Cross-functional teams are an important aspect to concurrent engineering. One critical example is the change control team. The change control team comprises many different personnel. System engineers help to diagnose and resolve hardware issues. Software personnel focus on software-related issues. Additional personnel keep track of process management, configuration management, documentation, and subcontractor management. Together these teams track progress within an integrated master schedule. This schedule maintains the team milestones, combining the individual schedules into one project schedule. Process is another identifying characteristic of concurrent engineering. This paper will look at a real example of software design as related to concurrent engineering.

Process is an important aspect to software design, so the process is documented within desk instructions. The Software Development Plan (SDP) outlines the overall development process. It summarizes the design milestones such as the Preliminary Design Review (PDR) and the Detailed Design Review. The appropriate artifacts are called out. Other desk instructions are pointed to for more detailed information. One such desk instruction is the Software Configuration Management Plan (SCMP). The SCMP covers the correct usage of the software change management software, Rational Clearcase. It documents the correct branch naming

conventions and covers the correct labeling for each phase; that is, labels are given for deliveries between subcontractor to prime contractor within the same software component, labels are given for deliveries between the software component to the software integration team (SWIT), and the labels are given for deliveries between the SWIT and the system test and evaluation (T&E) team.

Artifacts included in the design process according to the SDP are architectural decisions such as which algorithms are used, throughput estimations, and fault management. Where necessary, trade studies may be conducted to analyze various options. Modeling is done using Rational Rose. Important diagrams include sequence diagrams, interface class diagrams, thread definition and interaction diagrams, and state diagrams. Diagram conventions and common rules are specified in a Rational Rose Usage Desk Instruction. XML and other configuration data files are generated. A Software Design Document (SDD) covers the high-level design decisions for the software component. Message Lists and IDL are generated for interactions among the various software components.

As per the SCMP and SDP, each artifact is labeled and delivered to SWIT for each software baseline. There is a Peer Review in which all applicable teams are invited to review the artifacts. Other software components, software quality evaluators, testers, system engineers, and customer representatives are invited to these reviews. Findings are generated to point out deficiencies, mistakes, or questions. The software component team then works to resolve any deficiencies in order to close out the peer review process.

Each software component is baselined and submitted for testing by SWIT. Following successful testing, the overall software is then baselined as per the SCMP. This new overall software baseline is documented and prepared for delivery to the T&E group. The T&E group then does additional system-level testing.

The above software design approach documented several aspects of concurrent engineering. There is an integrated product management which includes an integrated master schedule tying together cross-functional teams. Various teams are allowed to concurrently develop their artifacts. Incrementally, baselines are set and shared among teams. Throughout the design process, the customer is involved. In the end, these processes lead to a realized system which is delivered to the customer.