

IV&V of Design Consistency

The Need for Design Consistency

“Verification is:

(A) The process of evaluating a system or component to determine whether the products of a given development phase satisfy the conditions imposed at the start of that phase.

(B) The process of providing objective evidence that the software and its associated products conform to requirements (e.g., for correctness, completeness, **consistency**, accuracy) for all life cycle activities during each life cycle process (acquisition, supply, development, operation, and maintenance); satisfy standards, practices, and conventions during life cycle processes; and successfully complete each life cycle activity and satisfy all the criteria for initiating succeeding life cycle activities (e.g., building the software correctly).”

-IV&V Technical Framework Revision N

So what does IV&V check for design consistency, and what are some examples of consistent and inconsistent design features?

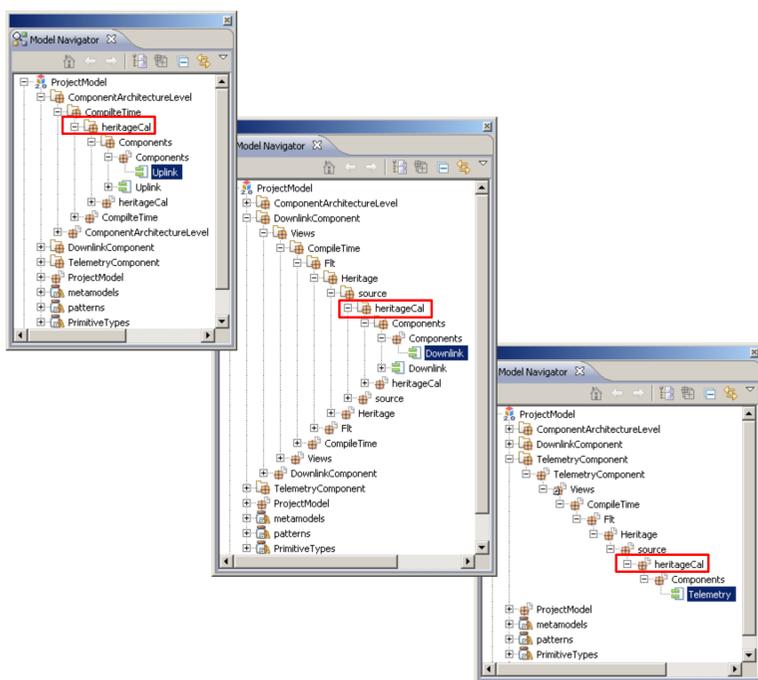


Figure 1: Design Structure Inconsistency

Figure 2, a component diagram, shows that two different hardware components are providing the same software interface. Figure 3 shows a sequence (interaction) diagram that goes into more detail than figure 2. While color coded with Figure 2 it does not utilize the same naming convention. This inconsistency while visually apparent is also pointed out in an audit report that captures and summarizes these type of inconsistencies.

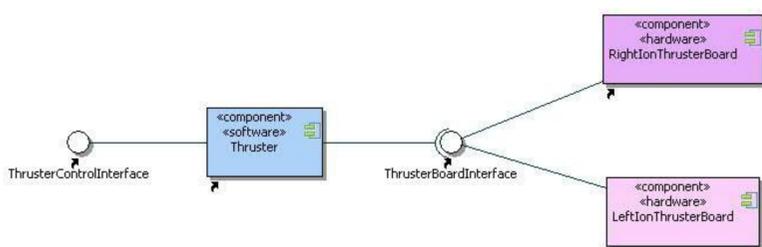


Figure 2: Interface Consistency



Source: www.spacetoday.org

Methods We Use

Checking the consistency of software code is not a simple task. It requires a thorough understanding of the developer’s required system functionality as well as a well-defined method for bringing subtle design inconsistencies to the attention of the analyst.

Sometimes no tools at all are needed to notice an inconsistency in the design of a product. An example of this can be seen in Figure 1. This figure consists of three snapshots of the locations of three different mission-critical software components within the developer’s design. Observe that the software components all come from the “heritageCal” package boxed in red. However, this package is retrieved from a different location for Uplink Component than it is for Telemetry and Downlink Components. This inconsistency creates an ambiguity about where the component data in “heritageCal” actually resides, allowing for convoluted code and unnecessary duplicates of model elements.

Aside from simple observation, one very useful tool IV&V uses to accomplish its more in-depth design consistency analysis is a UML-based platform that helps with defining precise relationships and responsibilities among individual model elements. By utilizing features provided by the UML platform, different types of diagrams can be computationally compared with one another to check for consistent uses of component-provided classes, operations, attributes and interfaces. This can aid in the check for consistency of the underlying code that would be generated by the UML model.

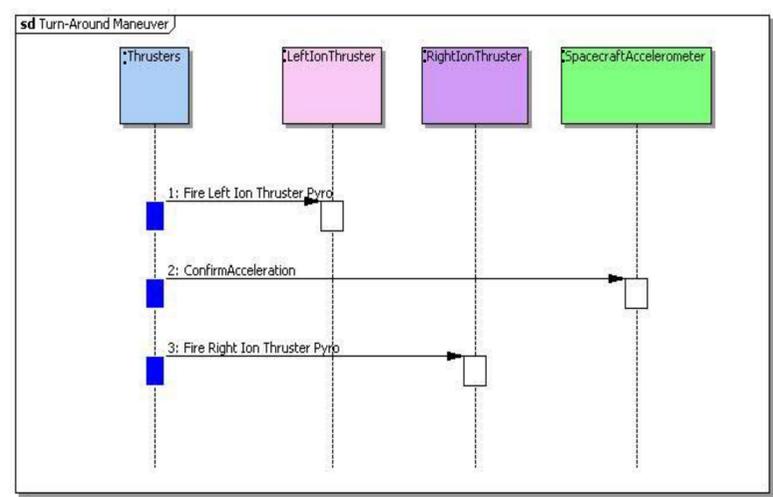


Figure 3: Design Inconsistency