4-1-2007

Systems architecting heuristics for systems engineering management and embedded systems engineering

Mark S. Anderson

Cihan H. Dagli
Missouri University of Science and Technology, dagli@mst.edu

Follow this and additional works at: http://scholarsmine.mst.edu/faculty_work

Part of the Operations Research, Systems Engineering and Industrial Engineering Commons

Recommended Citation
http://scholarsmine.mst.edu/faculty_work/2057

This Article - Conference proceedings is brought to you for free and open access by Scholars' Mine. It has been accepted for inclusion in Faculty Research & Creative Works by an authorized administrator of Scholars' Mine. For more information, please contact weaverjr@mst.edu.
Abstract - Software development for United States Air Force (USAF) weapon systems is a “right here, right now” prize captured by those who can rapidly develop requirements and deliver a quality product. The Air Logistics Centers (ALCs) located at Tinker Air Force Base (AFB), Oklahoma, Warner-Robins AFB, Georgia, and Hill AFB, Utah develop requirements utilizing 3400 funding to capture this prize. The ALCs identify these requirements as corrective maintenance or perfective and adaptive maintenance. Colleen A. Calimer and John L. BeVier introduce the concept of the “Embedded Systems Engineer” in their 2004 INCOSE paper “Embedded Systems Engineering: Managing Systems Complexity, Change, and Crises”. They present the case that insightful management and utilization of the “embedded” systems engineer is the critical component of successful engineering activity in the dynamic systems environment. This paper applies systems architecting heuristics to systems engineering management to suggest that “embedded” systems engineering is functionally beneficial to the operational demands of maintaining the capability of the Avionics Flight Software (AFS) of the B-1B strategic bomber weapon system.

DISCUSSION

In response to the challenges of the Global War on Terror, the USAF war fighter needs new technology and new capability as quickly as possible. Software development for USAF weapon systems is a “right here, right now” prize captured by those who can rapidly develop requirements and deliver a quality product. Software engineering development is best served by corrective maintenance or perfective and adaptive maintenance 3400 funding – funding managed by the ALCs located at Tinker AFB, Oklahoma, Warner-Robins AFB, Georgia, and Hill AFB, Utah. Success in this environment is the result of a paradigm shared among industry systems engineering management, ALC acquisition professionals, and the war fighter at Air Combat Command (ACC).

Maintaining the AFS computer software configuration item (CSCI) for a strategic weapon system such as the B-1B bomber affects tens of thousands of ADA source lines of code (SLOC) for each software cycle (called a sustainment block) of an avionics system that has hundreds of thousands of ADA SLOC. The AFS is an embedded system – described by Boehm as a “project operating within (is embedded in) a strongly coupled complex of hardware, software, regulations, and operational procedures...” The AFS is also a reactive system that may use physical sensors such as: defensive system line replaceable units, phase-arrayed radar, and navigation systems. B-1B AFS systems engineers use functional flow block diagrams (FFBD) to build an integrated model of the avionics system and support requirements development. In addition, they use “screen-shots” to depict where and how requirements affect the aircrew displays. This is a Hatley-Pirbhai (H/P) approach that allocates customer requirements to the system model. [1] The system model is highly
flexible and it enhances communication with the war fighter. This enables the customer to continually evolve the implementation of a requirement or function. H/P data-flow diagrams and “screen-shots” also communicate the functional decomposition of the requirements to the software designer. To this end, systems engineering serves a vital function in translating customer requirements into a framework for successful software design. Colleen A. Calimer and John L. BeVier introduce the concept of the “Embedded Systems Engineer” in their 2004 INCOSE paper “Embedded Systems Engineering: Managing Systems Complexity, Change, and Crises”. They present the case that insightful management and utilization of the “embedded” systems engineer is the critical component of successful engineering activity in the dynamic systems environment. It is an environment where “requirements... continually change and that those changes must be immediately accommodated so that the system can remain dynamic.” [2]

In their paper, BeVier and Calimer re-address the function of the systems engineering manager and define the concept of the “Embedded Systems Engineer”. They state that “the purposeful inclusion and redefinition of the systems engineer from that of a specialist supporting engineering activity to that of the persistent critical function underlying the management and direction of a project” is the most important function of the systems engineering manager. [3] For example, determination of scope defines the requirements for each “block” of software development for the B-1B. ACC representatives differentiate between “needs” and “desires”. Then ACC communicates these requirements to the acquisitions professionals at the ALC, and the Boeing systems engineering manager who directs the integrated product team (IPT). For an Air Force 3600 funded engineering, manufacturing, and development program, ACC normally documents its requirements in an operational requirements document (ORD). For the quicker paced 3400 funded programs, an engineering assignment (EA) documents ACC’s requirements; and the systems engineering manager has the responsibility to ensure that systems engineering and software design elements develop and deliver a quality product that meets the performance need of the war fighter (ACC) within the additional critical elements of cost and schedule. This responsibility demands that the systems engineering manager designate or utilize an existing “embedded systems engineer” to participate in the process from the onset. In the B-1B AFS environment, an “embedded” systems engineer supports the functions of requirements definition, contract development and implementation, program planning, scheduling, systems development, software design, and system test validation/verification.

The process of software development for corrective maintenance or perfective and adaptive maintenance “sustainment blocks” for the strategic weapons systems occur in a yearly cycle on the B-1B, but varies for other weapon systems. Each B-1B software block usually starts with a series of technical interface meetings (TIM) between systems engineers and the Air Force. Here is where the role of an “embedded” systems engineer is paramount. The first TIM is very high level – the major Air Force customers in this meeting examine a list of proposed corrective, or perfective and adaptive maintenance change candidates. Representatives from the ALC and ACC discuss their needs regarding each proposed change candidate. The ALC submits the list to ACC to “rack and stack”; and prioritize the items to the “mission”. The second TIM increases the involvement of the war fighter, while the “embedded” systems engineer supports the decision-making process by providing preliminary requirements definition and interface identification for each candidate. Then a third TIM occurs to present a last opportunity for the war fighter to look at the requirements. An “embedded” systems engineer supports this meeting with a product defined by systems engineering and software design. The TIM material includes charts displaying probable avionics display changes for each applicable item on the list. This visual presentation aids the decision making process and allows initial feedback from the operational community. ACC selects the items for the software block and the ALC releases the EA to Boeing. Several weeks prior to a preliminary design review (PDR), a crew station working group (CSWG) meeting occurs where systems engineers and software design review all display changes and human factors impacts with the ACC war fighter. In some instances, format display simulations using code prototypes are available to review at the crew stations in the avionics test laboratory. A second CSWG occurs prior to a critical design review (CDR). The systems engineering manager and an “embedded” systems engineer lead these events: three TIMs and two CSWGs with initial and preliminary
design prototypes. These presentations enable a review by the ALC and the war fighter in an environment that encourages feedback to systems engineering.

BeVier and Calimer state that this is an environment where the customer and systems engineering management should expect “requirements......to change during the life cycle of the system”. [4] It is vital to the life cycle of the strategic weapons system; and it meets operational need. Furthermore, BeVier and Calimer believe that these are “standing business process” by the “operational engineering arm” to encourage changes (are) made to the (strategic weapon) system during “operational employment.” [5] These initial major milestones for the project – PDR and CDR - are crucial meetings with the ALC and ACC for the systems engineering, and software design disciplines. At the PDR meeting, the ALC can trace how the CSCI meets the requirements. ACC’s requirements are captured within documents developed by systems engineering for example: a systems segment specification (SSS) document defines the overall environment of the embedded AFS; a systems requirements specification (SRS) document defines what the software must do; a software design document (SDD) defines specific requirements. A man-machine interface (MMI) document provides a home for updated “screen-shots” and captures the requirements for the ALC, the war fighter, systems engineering, and software design. The systems engineers who author this document deliver one of the most important products for the CSCI, since over 80% of each software block update involves corrective, or perfective and adaptive maintenance of an aircrew display. Finally, an interface control document (ICD) defines where the CSCI borders other CSCIs within the overall weapon system. Software design provides insight into initial design products that complete the trace from the EA statement of work (SOW) to the SSS to the SRS to the SDD. Moreover, at the CDR meeting, the ALC and ACC representatives witness how software design continues to capture requirements in final software design folders.

So, for the B-1B sustainment program, the EA defines the scope of change for the software block; it is essentially the statement of work for the engineering project. The systems engineering manager knows that the EA defines the product – the AFS CSCI – for the effort. And within this CSCI, major functions such as navigation, weapons delivery, radar target identification, defensive protection, terrain-following, system management and avionics test operations define the computer software components (CSCs). This aggregation is an important concern for systems engineering development to “group elements that are strongly related to each other, separate elements that are unrelated”. [6] Interface definitions occur at the systems level of any strategic weapons system distinguishing avionics software systems requirements from for example, integrated test systems, electric multiplexing systems, offensive radar systems, navigation systems, communications systems, and defensive avionics systems. In the B-1B AFS, partitioning of these requirements then occurs where the requirements are grouped by function for example: defensive, navigation, radar, weapons delivery, terrain following, avionics test, and systems management. Further identification of the specific computer software unit (CSU) for implementation of the requirement occurs after communication and peer reviews between systems engineers and software designers. Software design transitions from the desktop environment; and moves to an integrated avionics laboratory for level one and level two testing. Code and unit test activity at the CSU level are the foundation of design decisions for the software design group. Computer software component (CSC) CSC-to-CSC integration testing ensures that the CSC interfaces build a complete CSCI. The software test group conducts functional test activity to validate that the CSCI meets all design decisions. Finally, formal qualification test (FQT) activity certifies the CSCI for delivery to the Air Force flight test community. An “embedded” systems engineer assists the systems engineering manager in tracking all of the planned changes and deficiencies from code and unit test to delivery of the CSCI to flight test. Delivery of the CSCI to the flight test community puts the product into the world of developmental test and evaluation (DT&E). After a successful DT&E, the product proceeds to force development evaluation (FDE) - the ‘graduation exercise’ conducted by operationally qualified crewmembers from ACC. It is important to learn as much as possible about the exit criteria for successful completion of this final milestone. Ultimately, success depends on whether the systems engineering manager and the “embedded” systems engineer remain aware and manage the impact of planned changes, unplanned deficiencies, and the primary factors which are important to the ALC, the war fighter, and Boeing - “cost”, “schedule”, “performance”
and “quality” throughout all of the phases of the software block.

Throughout the process of requirements development, design, and test the typical heuristic exists for embedded AFS: “the cost of removing a defect from a software system rises exponentially with the number of development phases since the defect was inserted”. [7] Engineers conduct peer reviews of software modules early in the code and unit design phase – discovery of defects during this phase are easy to correct. These defects are typically within the independent computer software component and do not typically affect the other components. Defects discovered during CSC integration may influence an interface between the software components and thus involve greater team resources (cost) to solve and correct. Defects that occur during either the DT&E or FDE flight phases involve the ALC and ACC in the decision-making process in addition to the systems engineering manager, the “embedded” systems engineer, and the systems and software leads. As the heuristic intones, this is extremely costly because the software at this point is an integrated and configured item that in some instances would require extensive or selective functional and formal qualification testing. Cost prohibitive defect corrections challenge the customer to defer the correction of the deficiency to a follow-on software block. However, in the dynamic environment of the Global War on Terror, the war fighter often needs most if not all deficiencies corrected to meet the operational need. It is within this paradigm that BeVier and Calimer define where the “embedded” systems engineer can significantly contribute.

Now “the number of problems encountered in development is inversely related to their magnitudes”. [8] Knowledge of this heuristic, is well-understood in the above stated waterfall paradigm where software is a sequential link of several cyclical phases, going from requirements definition to code and unit test to computer software component integration to functional test to formal test case to flight test. In most programs, and it is no different in the B-1B, any engineer in any of these phases can generate a notification of a deficiency in the code – referred to as a software change request (SCR). An SCR establishes the formal process for documenting planned (requirements) changes, and correcting problem deficiencies. Engineers enter the SCR in a defect database tracking system, and the customer reviews the database for most programs on a weekly basis. The systems engineering manager and the “embedded” system engineer assign priorities from one to five to each SCR with one being the most severe. In the B-1B, the definitions (taken from MIL-STD 498) of these priorities are:

- A priority one SCR is a problem that prevents the accomplishment of an operational or mission essential capability specified by baseline requirements; a problem preventing aircrew accomplishment of an operational or mission essential capability; or jeopardizes aircrew safety;
- A priority two SCR is a problem that adversely affects the accomplishment of an operational or mission essential capability specified by baseline requirements to degrade performance and for which an alternative workaround solution does not exist. Or adversely affects aircrew accomplishment of an operational or mission essential capability specified by baseline requirements so as to degrade performance and for which an alternative workaround is not known;
- A priority three SCR is a problem that adversely affects the accomplishment of an operational or mission essential capability specified by baseline requirements to degrade performance and for which an alternative workaround solution exists. Alternately, the deficiency adversely affects aircrew accomplishment of an operational or mission essential capability specified by baseline requirements to degrade performance and for which an alternative workaround exists. A priority three SCR is the most common – discovered throughout the development cycle;
- A priority four SCR is a problem that is an aircrew inconvenience or annoyance; or does not affect the required operational or mission essential capability; and finally,
- A priority five SCR designates all other problems and is usually reserved for document changes. [9]

As stated, within the B-1B AFS program a weekly discussion of SCR resolution occurs with the ALC. Within this context, one role of the systems engineering manager is to communicate the priority and the resolution of each SCR whenever necessary to the ALC and the war fighter. And here is a final observation: the logical agent for intensive discussion of each SCR, whether it is a deficiency, a planned (or unplanned) requirement change or a candidate for inclusion in a later software block is the “embedded” systems engineer. The “embedded” systems engineer is
the individual present throughout all phases of the AFS development cycle for the maintenance of a strategic weapons system. An “embedded” systems engineer fully supports the multiple tasks, roles and responsibilities of the systems engineering manager.

**SUMMARY**

As BeVier and Calimer note, “The number one requirement for any system is to accommodate mission change”. [10] For the war fighter and the acquisition community, the changing operational requirement demands upon the corrective maintenance or perfective and adaptive maintenance of strategic weapons systems are certainly dynamic. A successful systems engineering manager of a strategic weapons system such as the B-1B is aware of common systems architecture heuristics that support requirements development. It is a success that embraces the concept of the “embedded” systems engineer where the “emphasis is placed on the number one system requirement: to flexibly morph system changes in response to changes in the customer domain, the technological domain, and in the operational domain.” [11] “Embedded” systems engineering is functionally beneficial to the operational demands of maintaining the capability of the avionics flight software of the B-1B strategic bomber weapon system.

**REFERENCES**


