Workshop on Product Lines for Command-and-Control Ground Systems at the First International Software Product Line Conference (SPLC1) in Denver, Colorado, August 2000.

FSATS: An Extensible C4I Simulator for Army Fire Support[1]

Don Batory
Department of Computer Sciences
The University of Texas
Austin, Texas 78712
batory@cs.utexas.edu

Clay Johnson, Bob MacDonald, Dale von Heeder
Applied Research Laboratories
The University of Texas
Austin, Texas 78712
{clay, bob, drv}@arlut.utexas.edu

1 Introduction

Fire support is a command-and-control application that includes the detection of targets by forward observers on a battlefield, the exchange of messages between command outposts to assign weapons to attack the target, and the coordination of weapon firings by forward observers during the attack. Fire support is one of a number of domains that has been modernized by digital *Command, Control, Communications, Computer, and Intelligence (C4I)* systems that automate battlefield mission processing. *AFATDS (Advanced Field Artillery Tactical Data System)* is arguably the most sophisticated C4I system in existence, and provides the software backbone (message transmission, processing, etc.) for fire support for the Army [3].

Simulation plays a key role in U.S. Army testing and training. It avoids costs of mobilizing live forces, provides repeatability in testing, and allows force-on-force combat training without the liability. *FSATS (Fire Support Automated Test System)* is a system for testing AFATDS and other fire-support C4I systems. FSATS collects digital message traffic from command and control communication networks, interprets these messages, and stores them in a database for later analysis. FSATS can simulate any or all OPFACs (or *operational facilities*, such as forward observers, command outposts, and weapons)[1]. The subject of a test can be overall system performance, individual OPFAC performance, or system operator performance. Thus, FSATS can be used both in training Army personnel in fire support and debugging/testing AFATDS.

Our interest in product-line architectures stems from a desire to re-engineer the original implementation of FSATS, begun almost 10 years ago. FSATS must constantly support new capabilities while improving existing ones, yet the current system is plagued by *design fatigue* — a state of design where further evolution is both difficult and costly. Product-line architectures are appealing because they offer the ability to add new features and replace/revamp existing features by adding and replacing components. While it is the case that different versions of FSATS are not simultaneously needed, the ability to modify and update FSATS more easily and economically (i.e., replacing one variant of FSATS with another) *is* a major goal.

Our redesign of FSATS centered around the notion of mission types as FSATS building blocks. A *mission type* is the set of protocols that forward observers, command posts, and weapon systems follow when executing an instance of this mission. A mission type is characterized by the method of controlling the attack and the weapon system selected. An example is *When-Ready-Fire-For-Effect (WRFFE) Artillery*: when the forward observer identifies an enemy tank concentration, he relays his report to his commanders who select field artillery to attack that target. The forward observer, commanders, and weapon systems interchange messages according to strict protocols to coordinate the attack. Another mission type is *Time-on-Target (TOT) Artillery*: field artillery is requested to fire at a target so that all rounds land at the specified location and specified time. In all, there are over 20 different mission types. Extensibility of FSATS is largely determined by the ease with which new mission types can be added and existing mission types can be updated.

1. This work was supported by the University of Texas Applied Research Labs and the U.S. Advanced Research Projects Agency in cooperation with the U.S. Wright Laboratory Avionics Directorate under contract F33615-91C-1788.

Our product-line architecture defines a component for each mission type. A version of FSATS that supports mission types WRFFE-Artillery and TOT-Artillery will include the components for these types. In general, a version of FSATS that supports mission types T_1, T_2, … T_n includes components for these types. Thus, evolving FSATS by adding new mission types and updating existing types is now a matter of adding and replacing components. The original implementation did not support the componentization of mission types; new mission types could not be added without careful analysis of their effect on existing mission types. Our redesign disentangles the logic of missions so that individual mission types can be defined and debugged largely in isolation from each other. This has proven to be a significant advantage.

FSATS is inherently a distributed application, where programs that simulate a *forward observer (FO)*, his commanding *Fire Support Team (FIST)*, the FIST's commanding *Fire Support Element Brigade (FSE-Brigade)*, and so on execute at different sites. An interesting feature of our components is that they "cross-cut" programs. In Figure 1 below, each column encloses the code for programs simulating an FO, FIST, and FSE-Brigade. Each of these programs would correspond to a CORBA or DCOM component in a distributed application. However, our components are orthogonal to these designs and are shown in dashed outlines. When we add a new mission type to FSATS, the FO, FIST, FSE-Brigade, etc. programs are updated to contain the new protocols that they must follow to process instances of that mission type. Thus the composition of our components yields different OPFAC programs that understand the suite of mission types that were defined in those components.

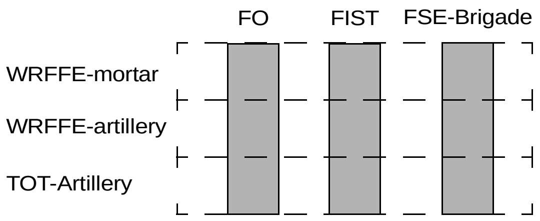

Figure 1: Cross-Cutting Effects of Mission Types vs. OPFAC Programs

2 Current Status

The redesign of FSATS was initially undertaken as a proof-of-concept to ascertain the applicability of a product-line architecture for producing a more maintainable distributed C4I simulation. Our success in that regard has led to a continuation of the work in which we have integrated the redesigned simulation with the existing FSATS communications components. It is now possible for the redesigned simulated OFPACs to interoperate both with live systems and with the legacy FSATS simulation. The next step is to add sufficient mission types to allow the redesigned simulation to replace the legacy system as the fielded version of FSATS. Further information on our product-line design is given in [2].

3 References

[1] "System Segment Specification (SSS) for the Fire Support Automated Test System (FSATS)", Applied Research Laboratories, The University of Texas, 1999. See also URL **`http://www.arlut.utexas.edu/~fsatswww/fsats.shtml`**.

[2] D. Batory, C. Johnson, B. MacDonald, D. von Heeder, "Achieving Extensibility Through Product-Lines and Domain-Specific Languages: A Case Study", *International Conference on Software Reuse,* Vienna, Austria, June 2000.

[3] "System Segment Specification (SSS) for the Advanced Field Artillery Tactical Data System (AFATDS)", Magnavox, 1999.