Case Study: Global Combat Support System - Air Force

Robert C. Seacord
Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213
+1 412 268-7608
rcs@sei.cmu.edu

ABSTRACT:
This case study describes the use of components to modernize legacy systems. The Global Combat Support System for the Air Force (GCSS-AF) seeks to modernize base level support systems into integrated systems that are responsive to Air Force needs during war and peace time. GCSS-AF mission areas include: Logistics, Finance, Personnel, Medical, Business Information and Functional Information Systems. The modernization efforts seek to maintain or improve current capabilities and reduce life cycle support costs without degrading current operations.

Keywords
Legacy systems, modernization, components, COTS.

1 BUSINESS PROBLEM

The GCSS-AF system is a system of automated information systems (AISs) requiring varying degrees of interoperability and information sharing [1]. A major focus of the GCSS-AF system is to specify and develop theses AIS as interoperable systems from conception. The overall goals of the program are to:

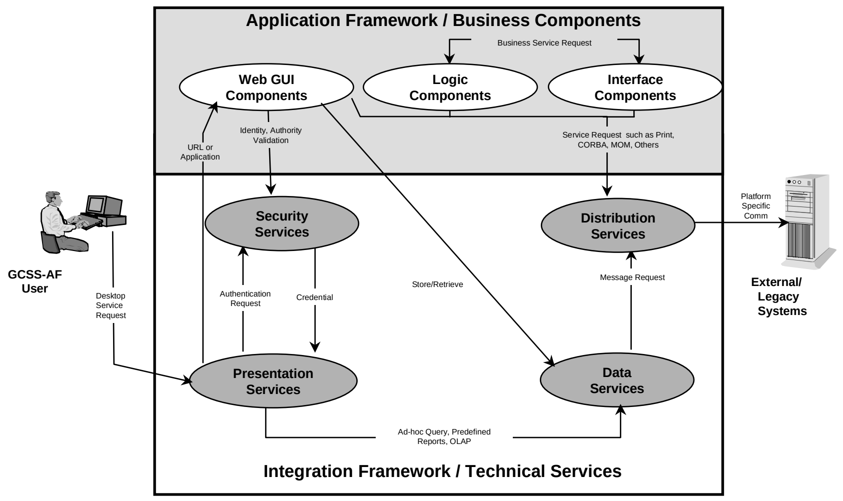
- make data available
- minimize AIS development, operation and maintenance costs

Essential to the development of an AIS is a robust, efficient, maintainable, extensible architecture and associated services. The benefits of rehosting these legacy applications to this framwork were viewed in terms of lower cost of ownership as a result of:

- Using component based architecture
- Reusing infrastructure and framework components
- Eliminating expensive bottlenecks
- Eliminating stovepipe operations and duplication
- Lowering maintenance costs (software and hardware)
- Leveraging enterprise licenses management
- Reducing interface management (via standardization)
- Implementing centralized system administration
- Realizing economies of scale across systems through improved efficiency of resources

2 TECHNICAL APPROACH

The GCSS-AF system architecture is used as a baseline for new AIS development and as an optimal end-state for modernization efforts. The GCSS-AF system architecture shown below, illustrates three major categories of components—Web GUI logic, and interface—and the primary information flows between these categories [2]. These components reside within the application framework and depend on technical services provided in the integration framework.



Services supporting the three technologies (CORBA, MOM and COM+) are provided in the integration framework layer. The application framework layer provides reusable business components and the business component framework that implements the mechanism for communications among business components. The frameworks define the architectural layers and capabilities that AIS developers use to implement their requirements. A developer must first define an AIS Architecture that is compliant with the GCSS-AF system architecture by selecting from the available services and business components available within the framework layers. If the required components are not available, then additional components are defined either as unique AIS components or common components to be added to the appropriate framework.

The definition of software component is used in GCSS is based closely on Clemens Szyperski definition [3] in that a software component is a unit of composition with contractually specified interfaces and explicit context dependencies only. A software component can be deployed independently, is self-contained, and is sufficiently specified to be usable by third parties.

The unit of composition for a component is defined by the content of a component and the results of analysis of the factors that contribute to the definition of the component. The contents of a component may include class libraries (C++, Java, 4GL); encapsulated software modules (ActiveX controls, JavaBeans, CORBA services); framework environments (OAG's OAMAS, IBM's San Francisco); CASE models; and prebuilt (COTS, Legacy) applications.

3 PROJECT DESCRIPTION

Version 1.2 of the GCSS-AF integration framework was completed in October of 1999 and consisted of CORBA services; limited security including SSL, single-sign-on, role-based access and some PKI support; message oriented middleware (MOM); and data services included support for Business Object Documents (BOD) [4]. Version 2.0 of the framework is scheduled for completion in March 2000 and will add expanded data services and security services as well as integrating Enterprise JavaBeans.

An initial pilot project is underway to migrate a legacy system to the 1.2 version of the integration framework. This effort has not yet completed, but initial indications are that the modernization effort is quite costly.

4 BENEFITS

As the principle goals of the componentization and modernization effort are to make data available and to reduce maintenance costs it is too early to demonstrate any benefits from the effort.

5 LESSONS LEARNED

The existing GCSS integration framework includes both CORBA and MOM services. In ensuing versions of the framework both EJB and COM+ will be added. Each of these middleware components provides overlapping services. Determining which services to use, and how best to use and combine these varying middleware services is a necessary part of the design process. We have found that the use of model problems to learn about products in a particular problem context is essential to the design process.

The cost of modernizing the legacy system has proven to be quite high. In general, it is not possible to find developer who have experience in both the technology used in the legacy system and in the technologies dictated by the use of the integration framework. It is prohibitively expensive to maintain the right kind and level of competence in the face of new and emerging products and technologies. Competence must be developed when and where it is needed.

The integration framework is a moving target. As large areas of the framework were undefined during the development of the project pilot, decisions had to be made "on the fly" as to how these services would be provided or problems addressed. Having identified these gaps, the integration framework team can now best consider how to resolve these problems. Unfortunately, if these solutions are different from the ones adopted by the pilot project the "modernized" system would need to be retrofitted to these changes or remain non-conforming with the GCSS-AF infrastructure.

6 REFERENCES

1. GCSS-AF System Requirements Specification, Version 2.0 Document No. GCSS-REQ-1997-0001.
2. GCSS-AF Architecture Overview, Version 2.1 Document No. GCSS-REPORT-1997-0010.
3. Component Software Beyond Object-Oriented Programming, Clemens Szyperski, Addison-Wesley, 1998l
4. White Paper: Plug and Play Business Software Integration The Compelling Value of the Open Applications Group, Open Applications Group, July 1999.