Global Combat Support System - Air Force

Architecture Overview and Description

DRAFT

Prepared for:
Department of the Air Force
Headquarters Standard Systems Group (SSG)
Maxwell Air Force Base - Gunter Annex
Montgomery, Alabama

Contract Number: **F01620-96-D-0004**

Document Number: **GCSS-REPORT-1997-0010**
Version: **3.1**
Date: **07/28/00**

Lockheed Martin Federal Systems

1801 Route 17C

Owego, NY 13827-3998

This Page Intentionally Left Blank.

DOCUMENT CHANGE HISTORY					
VERSION			CHANGE DESCRIPTION (REQUIRED FOR CHANGES AFFECTING TPM, REQUIREMENTS, CONFIGURATION, COST & SCHEDULE)		
NO.	APPROVAL	DATE	CHANGE DOC.	SECTION	NARRATIVE (OF ITEMS AFFECTED)
1.0	**CCB**	**12/12/97**			**INITIAL RELEASE**
2.0	**CCB**	**02/08/99**		**ALL**	**SCR 059 – INCORPORATE INTEGRATION FRAMEWORK & COMPONENTIZATION.** **UPDATES FOR CORE AND SSTR.**
2.1	**CCB**	**06/09/99**		**ALL**	**SCR 097- INCORPORATE CUSTOMER COMMENTS.** **UPDATES FOR CORE SECURITY RELEASE.**
2.2 DFT A	**IPT**	**09/22/99**		**ALL**	**INCORPORATE MOM AND XML FOR BODS FOR IF 1.2 RELEASE. (TASK ORDER 5075)**
2.2	**CCB**	**03/13/2000**			**SCR 0111 – UPDATES FOR IF RELEASE 1.2**
3.0 DRAFT	**IPT**	**07/19/2000**		**ALL**	**CR-0009-CHANGE TITLE FROM GCSS-AF DEVELOPER'S GUIDE TAB A – SYSTEM ARCHITECTURE OVERVIEW AND DESCRIPTION TO GCSS-AF ARCHITECTURE OVERVIEW & DESCRIPTON** **MAJOR RESTRUCTURING OF THE DOCUMENT**
3.1 Draft		**7/28/2000**		**ALL**	**GENERAL CLEAN-UP: EXPANDED ACRONYMS FOR CLARITY, ADDED MISSING PARENTHESES AND UPDATED FIGURE 4**

This Page Intentionally Left Blank.

DRAFT

TABLE OF CONTENTS

DRAFT

LIST OF FIGURES

DRAFT

LIST OF TABLES

1. Introduction

GCSS-AF provides a component-based Reference Architecture Framework that serves as the Integration and Application Framework Layers for GCSS-AF functional capabilities consistent with the Defense Information Infrastructure Common Operating Environment (DII COE), the Joint Technical Architecture - Air Force (JTA-AF), and based on commercial open standards. The GCSS-AF Reference Architecture Framework also provides common interfaces for those functions that either directly or indirectly support Command and Control (C2) or share information with C2 Systems.

It is assumed that the reader is cognizant of Object Oriented Analysis and Design methods, the Unified Modeling Language (UML), Open Applications Group (OAG) concepts and specifications, Object Management Group (OMG) concepts and specifications, and GCSS-AF Requirements Specifications. Specific reference documents are provided in Section 2.

1.1 Purpose

1.1.1 Overview

Application Developers associated with GCSS-AF will be performing development in a new environment with new processes, techniques, and constraints. Guidance is needed to understand the overall integration environment. This document is one of an interrelated set of four primary sources of information for developing applications within GCSS-AF:

A. Global Combat Support System - Air Force (GCSS-AF) Architecture Overview and Description
B. Global Combat Support System - Air Force (GCSS-AF) Application Framework Developer's Guide
C. Global Combat Support System - Air Force (GCSS-AF) Guide to Developing with the GCSS-AF Integration Framework
D. Global Combat Support System - Air Force (GCSS-AF) Systems Solutions UML Model

In addition, there should be a Developer's Guide unique to the Business Area under development. The document and model interrelationships are depicted in Figure 1.

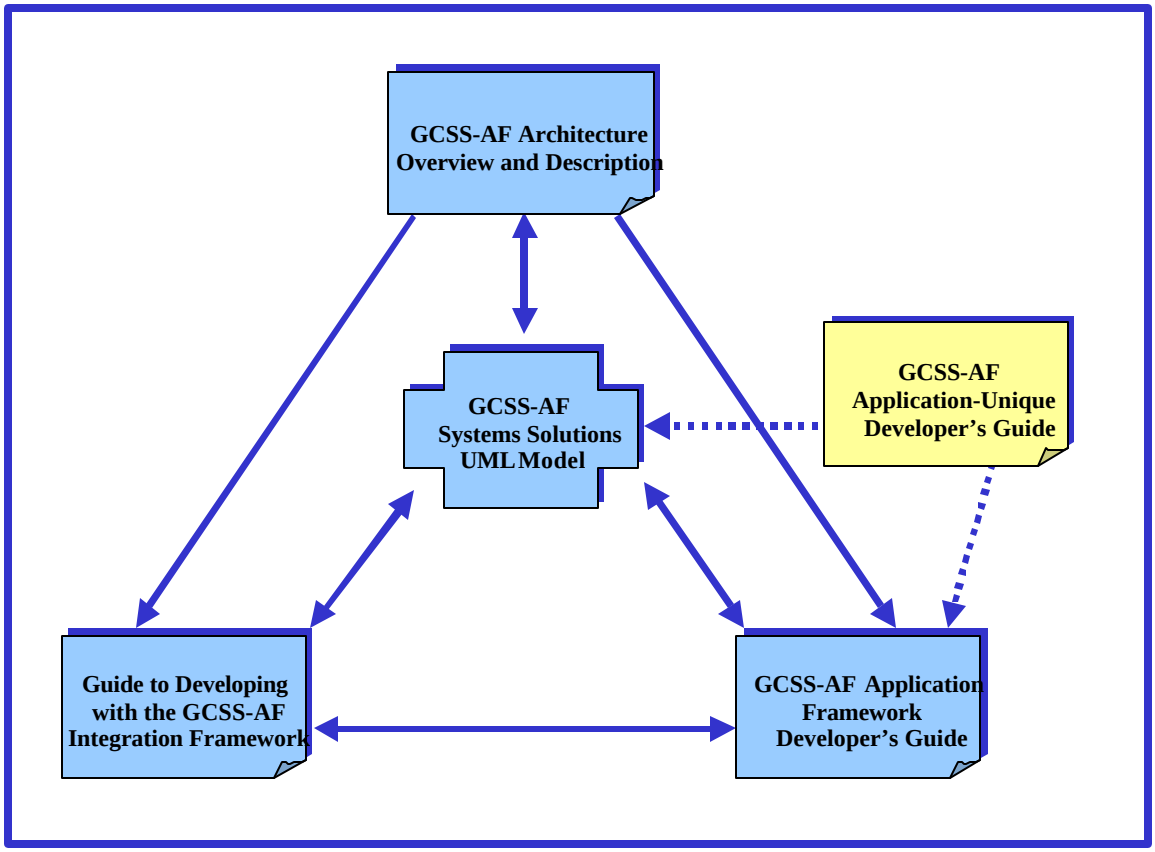

Figure 1 GCSS-AF Document and Model Inter-relationships

For the Application Developer to obtain a complete understanding of the definitions, concepts and processes, the information above should be read/used in the order above (A through C with references to D, as necessary) and sequentially within each document to build a complete understanding of the development methodology.

In addition, the Application Developer should review the following documents to understand the overall GCSS-AF Requirements as well as the specific Integration Framework requirements:

E. Global Combat Support Systems – Air Force System Requirements Specification
F. Global Combat Support System - Air Force (GCSS-AF) Integration Framework Enterprise Systems Management (ESM) Requirements Subsystem Specification
G. Global Combat Support System - Air Force (GCSS-AF) Integration Framework Security Requirements Subsystem Specification
H. Global Combat Support System - Air Force (GCSS-AF) Integration Framework Data Warehouse Services Requirements Subsystem Specification

1.1.2 Purpose of this Document

The purpose of this document is to provide an overview and description of the Global Combat Support System - Air Force (GCSS-AF) Architecture. The architecture will be described in terms outlined in the Command, Control, Communications, Computer, Intelligence, Surveillance, and Reconnaissance (C4ISR) Architecture Framework which defines an architecture in three views: Operational Architecture View, Systems Architecture View, and the Technical Architecture View. The GCSS-AF Architecture Overview & Description, this document, the GCSS-AF Application Framework Developer's Guide, and the Guide to Developing with the GCSS-AF Integration Framework address only the System Architecture View of GCSS-AF.

This document, GCSS-AF Architecture Overview and Description, provides the Reference Architecture overview and describes the Application and Integration Frameworks and their relations to the Global Combat Support System - Air Force (GCSS-AF) Reference Architecture and the associated GCSS-AF Systems Solutions UML Model.

1.2 Objectives

This document presents an overview of the GCSS-AF Reference Architecture and its constituent parts. It is intended for Program Management and Lead Architects associated with GCSS-AF system modernizations and owners of existing Air Force applications that participate in the accomplishment of the current Air Force combat support operational capability. This document serves as a pointer to the set of associated documents which provide the detailed descriptions of the GCSS-AF Reference Architecture for Mission Application Developers who need to understand the Reference Architecture at a detailed level.

1.3 Scope

This document is composed of the following sections:

Section 1: This section provides an Introduction, Purpose and Scope of this document.

Section 2: This section contains lists of Reference Documents.

Section 3: This section provides an overview of the GCSS-AF Architecture by identifying the purpose of GCSS-AF and the tenants of Commercial-off-the-Shelf products and commercial standards. This section also provides a high-level view of the Reference Architecture and the associated layers.

Section 4: GCSS-AF System Solutions UML Model: Provides a high-level description of the GCSS-AF model. It describes the overall model structure as well as the Development Life Cycle.

Section 5: Architectural Concepts: Identifies the concepts that form the foundations of the GCSS-AF Architecture.

Section 6: The Future of the GCSS-AF Reference Architecture: Identifies the time phased transition from the way applications are currently implemented to the target GCSS-AF Architecture for various architectural elements.

For a list of Acronyms and Glossary of Terms, reference the GCSS-AF Developer's Guide – Architecture Dictionary and Acronyms; GCSS-REPORT-1999-0100.

2. Reference Documents

- Defense Information Infrastructure (DII) Common Operating Environment (COE) Integration and Run-time Specification (I&RTS), Version 3.0.
- C4ISR Architecture Framework, CISA-0000-104-96, Version 2.0.
- Global Combat Support System - Air Force (GCSS-AF) Architecture Overview and Description GCSS-REPORT-1997-0010.
- Global Combat Support System - Air Force (GCSS-AF) Application Framework Developer's Guide, PROJ-2000-GCSSAF-0371.
- Global Combat Support System - Air Force (GCSS-AF) Guide to Developing with the GCSS-AF Integration Framework, GCSS-REPORT-1997-0011.
- Global Combat Support System - Air Force (GCSS-AF) Systems Solutions UML Model
- Global Combat Support System - Air Force (GCSS-AF) System Requirements Specification; GCSS-REQ-1997-0001.
- Global Combat Support System - Air Force (GCSS-AF) Integration Framework Enterprise Systems Management (ESM) Requirements Subsystem Specification, GCSS-SPEC-1999-0110.
- Global Combat Support System - Air Force (GCSS-AF) Integration Framework Security Requirements Subsystem Specification, GCSS-SPEC-1999-0111.
- Global Combat Support System - Air Force (GCSS-AF) Integration Framework Data Warehouse Services Requirements Subsystem Specification, GCSS-SPEC-1999-0112.
- Global Combat Support System - Air Force (GCSS-AF) Developer's Guide – Architecture Dictionary and Acronyms; GCSS-REPORT-1999-0100.
- Global Combat Support System - Air Force (GCSS-AF) Guide to GCSS-AF Compliance; PROJ-2000-GCSSAF-0315.
- Component Software *Beyond Object-Oriented Programming*, Clemens Szyperski, Addison-Wesley, 1998
- Object Management Group (OMG) Outlines CORBA 3.0 Features, OMG's Press Releases, 9 Sep 1998.
- The Open Applications Group Worldwide Web Site: http://www.openapplications.org

- Open Applications Group Middleware API Specification, Revision 1.0, Document Number OAMAS_990815, Open Applications Group, 15 Aug 1999.

- Open Applications Group Integration Specification, Release 6.2, Document Number 991110, Open Applications Group, 10 Nov 1999.

- Unified Modeling Language (UML), Rational Software Corporation, http://www.rational.com/uml/resources/index.jtmpl

- Visual Modeling with Rational Rose and UML, Terry Quatrani, Addison Wesley Longman, Inc., 1998.

3. GCSS-AF Architecture

3.1 Reference Architecture Overview

The Layered System View of the Reference Architecture Model, shown in Figure 2 - GCSS-AF Reference Architecture, is defined to support distributed component-based applications developed for a distributed environment. This architecture also provides a structure that will enable interfacing with the monolithic applications that exist as legacy systems as well as legacy client-server applications. Each layer of the Reference Architecture is built using capabilities from the layers below it as needed.

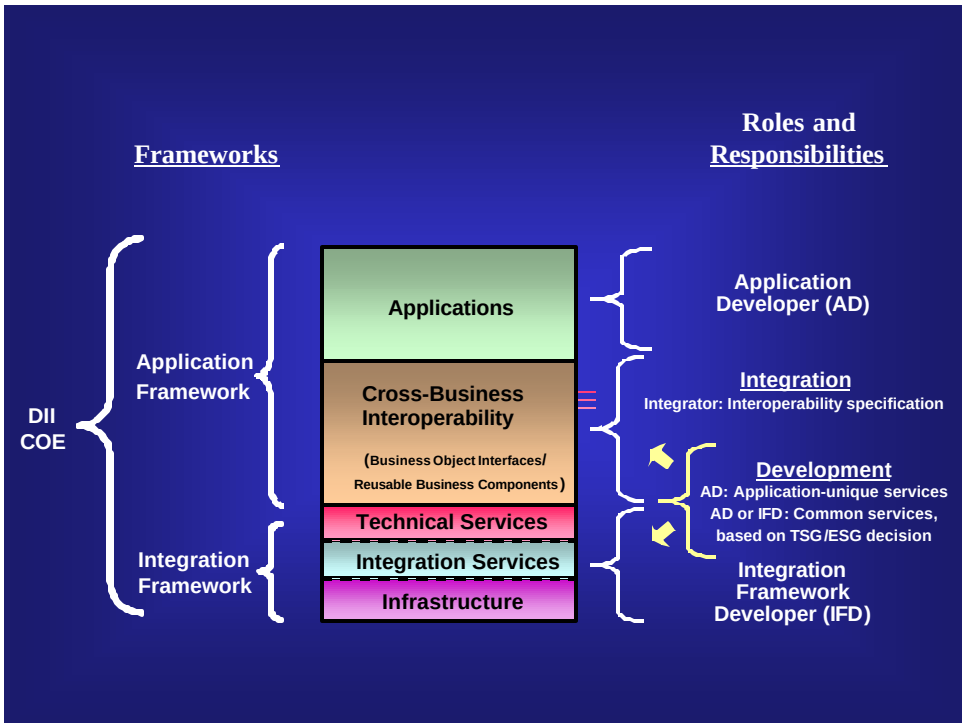

Figure 2 - GCSS-AF Reference Architecture

The GCSS-AF Reference Architecture is composed of 5 layers grouped into two major frameworks. The Integration Framework supplies the facilities and services that are utilized to build and execute mission applications and are DII COE level 6 compliant (with a goal of level 7). The Application Framework provides reusable business components and the business object interfaces that implement the mechanisms for communication among business components and should also be DII COE level 6 compliant (with a goal of level 7).

A Mission Application (MA) implemented using this Reference Architecture is composed of pieces of each layer starting at the bottom, building the system by identifying and using capabilities from each layer, in turn, to satisfy the system requirements.

The Roles and Responsibilities in Figure 2 - GCSS-AF Reference Architecture indicate which portion of the Reference Architecture is being/will be developed by the Integration Framework Developer (IFD), the Integrator, and the Applications Developer (AD). The IFD is responsible for all aspects of the GCSS-AF IF. The Integrator is responsible for the Interoperability Specification. The AD is responsible for all Application-unique aspects. If there are Application-unique services that are required for Cross-Business Interoperability, then the AD will develop that service. If there are common services in the Cross-Business area, then the Technical Steering Group (TSG) and Executive Steering Group (ESG) will decide on whether the IFD or AD will accomplish this development. For specific details on Methodology of GCSS-AF IF Compliance, the process for enabling GCSS-AF IF Compliance, Exception Conditions, and Application Validation and Integration, reference the Guide to GCSS-AF Compliance.

3.1.1 Integration Framework (IF)

The IF provides the foundation and building blocks upon which all GCSS-AF applications should be built. The availability of this foundation enables cost and schedule savings through shared use of developed and documented facilities and services and reduces the effort required to integrate modernized and newly developed systems.

The IF is composed of the Infrastructure, Integration Services and Technical Services Layers. The facilities and services provided by these layers are being centrally developed and implemented for the Combat Support community as the GCSS-AF IF. The current and future IF services are described in the Guide to Developing with the GCSS-AF Integration Framework. A summary of these layers of the architecture is provided below.

The lowest layer, **Infrastructure**, provides the Operating System (OS) and major system level COTS packages like the Database Engine. This layer also contains the hardware, such as clients, servers, Local Area Network (LAN)/Wide Area Network (WAN), network devices, and cabling.

Moving up to the next layer, **Integration Services** provides the communication protocols and methods such as CORBA, MOM or COM+ that are most often identified as Middleware.

The next layer is the **Technical Services** which provides distribution, presentation, data and security as well as enterprise system management services and facilities required to enable the construction of component based systems.

3.1.2 Application Framework

The Application Framework is composed of the **Cross-Business Interoperability** layer and the **Applications** layer. A summary of these layers is provided below. The capabilities of these layers are implemented via individual mission applications.

The **Cross-Business Interoperability** layer defines the cross business area and business area specific functional components and the associated data model. Business areas such as financial, logistics, personnel and medical are represented here. This level also defines and implements the rules for communications, interoperability capabilities and constraints among the Business Components within the architecture.

Finally, the **Applications** layer contains the typically coarse-grained Business Components, which implement the business logic that is specific and unique to the functionality being provided to the user. These components implement what is not available in any other layer. These components must also be DII COE compliant. This layer also encompasses the development tools required to construct and assemble components.

3.2 Detailed Reference Architecture Description

Figure 3 - GCSS-AF Reference Architecture (Second Level Decomposition) shows a more detailed allocation of technologies across the system view of the reference model. This section describes in more detail the layers of the architecture. Each level of the reference architecture would encompass the entire product set or a subset of the preceding levels. The following paragraphs describe various services and possible implementations. It should be stressed that not all of these are planned for inclusion in the Integration Framework at this time. The reader is referred to the Integration Framework Guide for a detailed list of the currently provided services.

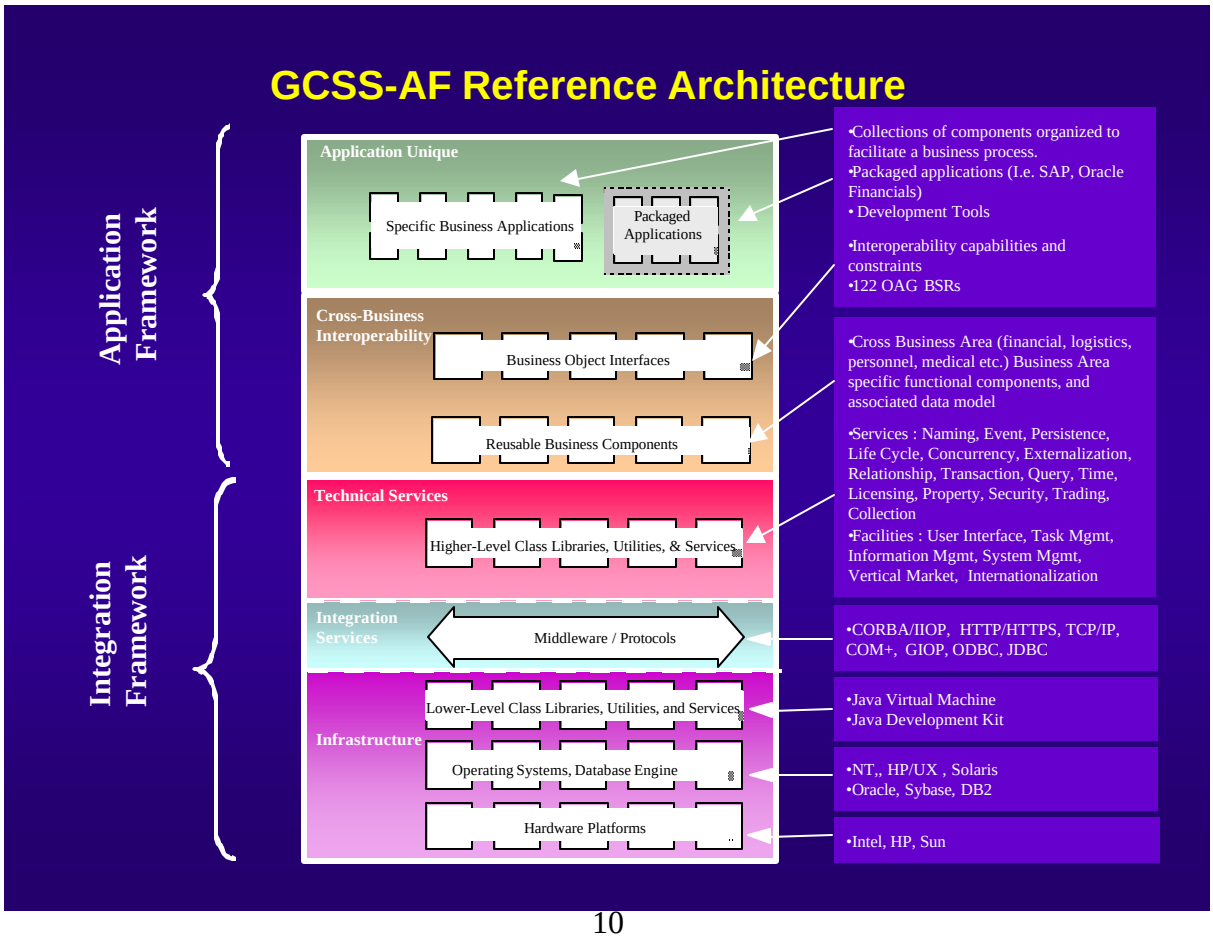

Figure 3 - GCSS-AF Reference Architecture (Second Level Decomposition)

3.2.1 Infrastructure

The Infrastructure Layer defines the baseline hardware and software technology required to enable rapid development and deployment of scalable component-based applications within performance, cost, and reliability constraints (e.g. operating systems, database engines, workstations, servers, networking hardware and software).

The bottom set of rectangles represents the Hardware Platforms of the distributed GCSS-AF Enterprise. The list reflects the set of DII COE compliant Hardware.

The next set of rectangles represents the Operating System and Database Engine environments required by distributed GCSS-AF applications. (Note that the layered approach as presented does not illustrate a division into client, server, or database tiers; actuality there will be hardware and operating systems chosen to support nodes in each of these tiers.) The GCSS-AF Enterprise will require different operating systems that can be mixed in a variety of ways. For example, you might use Windows NT and/or Windows 95 for client platforms; Unix or Windows NT for the middle tier; and Unix or Windows NT for the data tier. This architecture acknowledges that the specific operating systems used will become increasingly less important (as will the hardware on which the operating systems reside), and the browsers, Web servers and Internet protocols that support the distributed applications will be the key platform components.

The final set of rectangles in the Infrastructure Layer refers to basic services used by any application running on any platform. In a Java environment, this would include the Java Development Kit (JDK) classes and the Java Virtual Machine (JVM).

These products establish the hardware and system software baseline for the architecture being defined at the enterprise, framework or application level of the architecture hierarchy. This layer will consist of Defense Information Infrastructure (DII) Common Operating Environment (COE) compliant products.

3.2.2 Integration Services

The Integration Services layer defines the communications protocols and methods used to support communications among components as well as legacy systems. The integration services include distributed control, messaging, distributed data access mechanisms, and distributed communications protocols.

Distributed control mechanisms for synchronous component communications and control are provided through direct method calls. These protocols include CORBA Object Request Brokers (ORB) and Microsoft's component Object Model + (COM+). COM+ replaces Distributed Component Object

Model (DCOM) and is constrained to the Desktop (Client) for Windows 2000. ORBs are utilized for communication on and between Servers and possible at the Desktop (client).

Messaging protocols allow for asynchronous communication providing interfaces through messages rather than direct method calls. These protocols include Message Oriented Middleware (MOM) using the OAG's OAMAS specification. MOM allows for staging and prioritizing of work, controlling workflow in the receiving end without blocking the sending component if the receiving component is unavailable.

Distributed Data Access protocols connect applications to databases. These protocols include Microsoft's Open Database Connectivity (ODBC) and Java Database Connectivity (JDBC). ODBC provides a set of Call Level Interface APIs to SQL enable applications for relational databases. JDBC provides an object interface to SQL enable Java code for relational databases.

Distributed Communications protocols enable communication between the distributed services of the architecture. These include General Inter-ORB Protocol (GIOP), Internet Inter-ORB Protocol (IIOP), and Hypertext Transfer Protocol (HTTP). GIOP and IIOP are used for communications between ORBs. IIOP supports communication over a TCP/IP network and supports Java, JavaBeans, and Enterprise JavaBean intercommunications through an ORB. HTTP is used for communications between Web Browsers and Web Servers.

3.2.3 Technical Services

The Technical Services define the services and facilities required to support component based systems. The Knowledge (Metadata) Repository, allocated to this layer, provides the services necessary to capture, configuration manage, and access all of the architecture process artifacts including the reference architecture. These services encompass the services used by CORBA developers and these utilities would also include the component frameworks needed to support the Microsoft Transaction Server and Enterprise JavaBeans (EJB).

The services and facilities are identified from the OMG model but are not to be constrained to objects. Instead they are to be applied to components which may include objects but which have their own set of properties. In either case the same types of services are required although the specifications and implementation would be different.

Services standardize life-cycle management of objects. Interfaces are provided to create objects, to control access to objects, to keep track of relocated objects, and to control the relationship between styles of objects (class management). Also provided are the generic environments in which single objects can perform their tasks. Object Services provide for application consistency and help to increase programmer productivity. The 15 services are : Naming, Event, Persistence, LifeCycle, Concurrency, Externalization, Relationship, Transaction, Query, Licensing, Property, Time, Security, Trading and Collection. Details on these services are provided in the Guide to Developing with the GCSS-AF Integration Framework.

Facilities provide a set of generic application functions that can be configured to the specific requirements of a particular configuration. These are facilities that sit closer to the user, such as printing, document management, database, and electronic mail facilities. Standardization leads to uniformity in generic operations and to better options for end users for configuring their working environments. These facilities also include facilities for use over the Internet. The Facilities are: User Interface, Information Management, Enterprise System Management, Task Management, and Internationalization. Details on these facilities are provided in the Guide to Developing with the GCSS-AF Integration Framework.

3.2.4 Cross-Business Interoperability

The lower set of rectangles in the **Cross-Business Interoperability** Layer in Figure 3 - GCSS-AF Reference Architecture (Second Level Decomposition) represent reusable coarse-grained Business Components available for use by many specific business applications. The business area components are somewhat generic and are typically tailored, to be used as part of an application at the highest level.

The upper set of rectangles represent Business Object Interfaces. These are the rules for communications, interoperability capabilities and constraints among the Business Components within the architecture.

The Business Object Document (BOD), as defined by the Open Applications Group (OAG), is the interoperability model for specific Business Area message definitions. A BOD provides a standard message format, independent of mechanism for sending and receiving the message. The BOD's Control Area includes on of the OAG defined Business Service Request (BSR) which defines the action the sender wants the receiver to perform. The BOD's Business Data Area includes a definition of the data, making it a self-describing message format, and one or more occurrences of the data values. More detail on the OAG BODs and BSRs is located in the GCSS-AF Application Framework Developer's Guide as well as the OAG web site.

3.2.5 Applications Layer

The typically coarse-grained Business Components (e.g. implementations of the Business Objects) in the **Applications** Layer implement the business logic that is specific and unique to the functionality being provided to the user. These components implement what is not available in any other layer and rely on the services in the other layers.

Figure 3 - GCSS-AF Reference Architecture (Second Level Decomposition also depicts a representation of a packaged application – now usually called Enterprise Resource Planning (ERP) packages. A packaged application is a collection of business components tailored for a specific task. It typically includes its own data storage capabilities and may include generic business services or utilities. Note that while original ERP products were modularized and didn't support components as such, the

leading ERP vendors are currently working to break their packages up into components and to support the popular component standards. This is further exemplified in the corporate participation in the OAG. This layer also encompasses the development tools required to construct and assemble components.

3.3 Building an Application using the Architecture

The Guide to Developing with the GCSS-AF Integration Framework discusses how an actual application relies on a complete set of frameworks, services, databases, and platforms. Conceptually, an application is built at the top layer of the Reference Architecture and utilizes the underlying facilities and services. The key concept to building an application is that one must think of an application as a ***virtual*** collection of frameworks, services, data, and platforms. Another application may use many of the same resources depending upon the nature of those resources. A component-based application simply describes a set of components that work together to handle one type of problem; there is no hard boundary around the application as there was around conventional applications in the past.

4. GCSS-AF Systems Solutions UML Model

The GCSS-AF Systems Solutions Unified Modeling Language (UML) Model (The Model) represents the Architecture of GCSS-AF in an implementation independent manner. It provides opportunities for reuse, enables technology infusion, promotes a high-level systems view of the project, and enhances business modernization by promoting best business practices. The UML Model will form the foundation of the GCSS-AF Interface Repository.

The Reference Architecture is modeled in the System Solutions UML Model providing the operational and logical artifacts, such as Use Cases, Scenarios and UML Diagrams and mapping from the problem domain definition into the solution domain via the Component and Deployment Views. The Model enables navigation to pertinent pieces of the Reference Architecture.

4.1 Model Structure

The Model follows the Unified Software Development Process with extensions to accommodate the scope of Enterprise Development. Object-Oriented Analysis and Design techniques using UML syntax are employed to develop the GCSS-AF Reference Architecture. The Model representation follows three separate classification schemes.

1. **Program or Business view of the Model** - each Business Area or Program has a set named area where their particular artifacts are composed and viewed.
2. **Lifecycle view** - within the Business Area or Program all efforts go through the 5 phases of the GCSS-AF Lifecycle creating the appropriate model of that phase.

Layered View of the Reference Architecture - **to separate areas of concern over the layered approach framework of the Reference Architecture.**

Figure 4 - GCSS-AF Development Life Cycle and Figure 5 – GCSS-AF Reference Architecture Model Representation represent how a Program or Business Area would compose and view the various artifacts that model their solutions. Further details related to the classifications schemes are found in the GCSS-AF System Solutions UML Model itself, appropriate Program or Business Areas Developer's Guides, the GCSS-AF Application Framework Developer's Guide, and the Guide to Developing with the GCSS-AF Integration Framework.

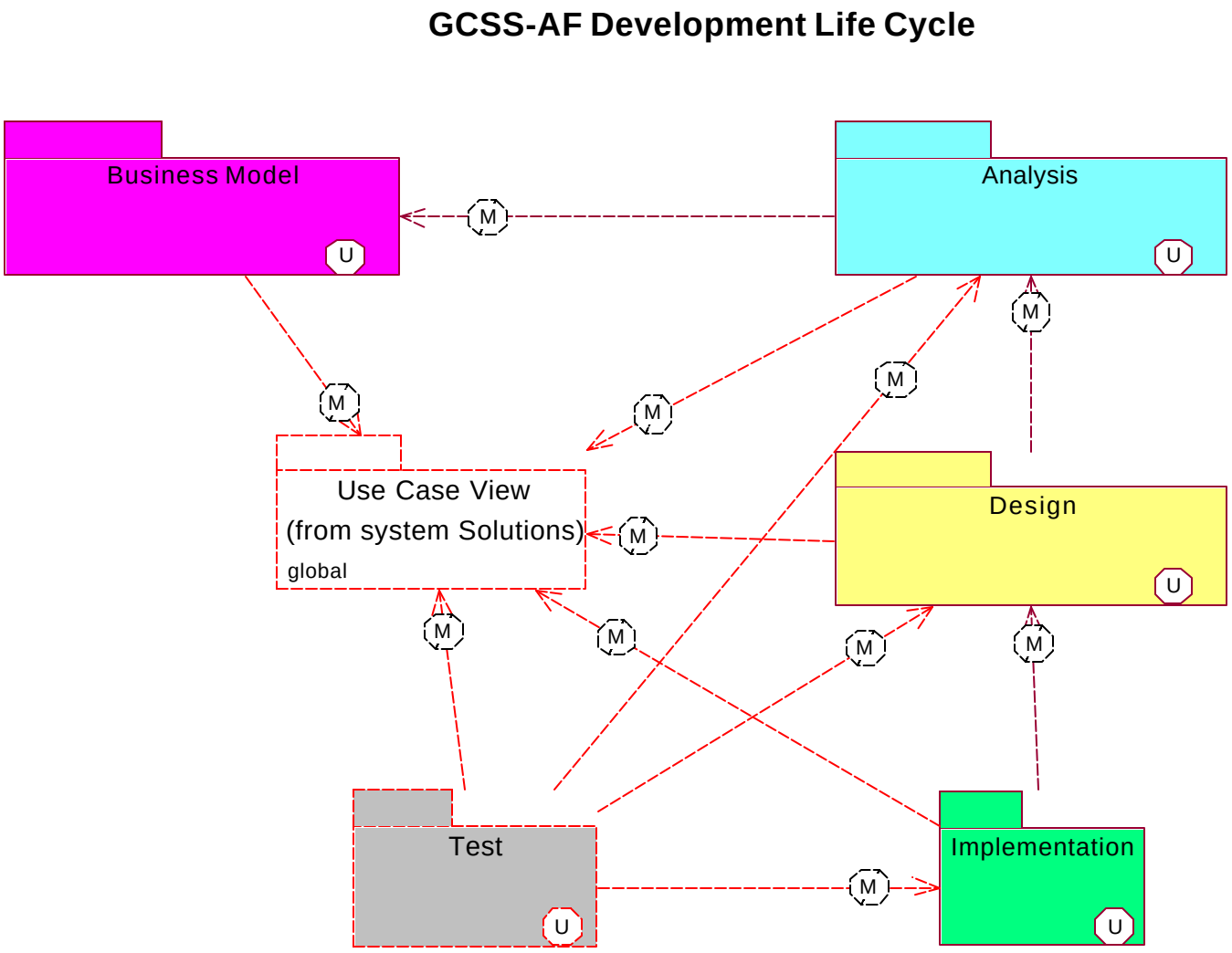

Figure 4 - GCSS-AF Development Life Cycle

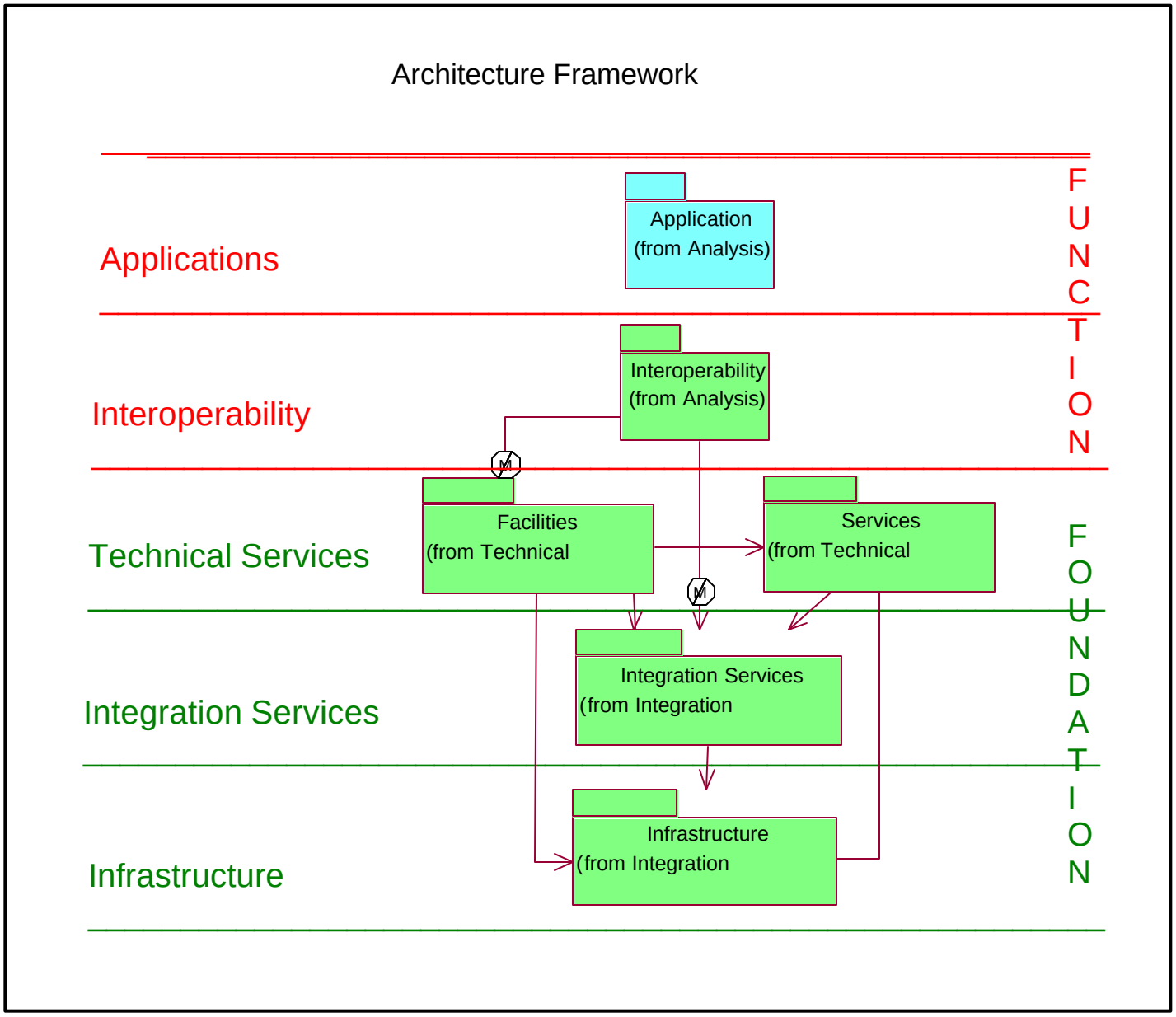

Figure 5 – GCSS-AF Reference Architecture Model Representation

5. Architectural Concepts

5.1 The Open Applications Group (OAG)

The tenants of the Application Framework are based upon the Open Applications Group (OAG) integration concepts. The OAG is a non-profit, vendor centric, consortium comprised of enterprise application software developers. The purpose of the OAG is to create Open Applications Integration by establishing and publishing specifications to enable business object integration across the enterprise. Their focus is on the concept of a Business Object Document (BOD) and its associated business processes used to exchange data between business components. Currently, there are two specifications within the OAG:

- Business – Open Applications Group Integration Specification (OAGIS)
- Technical – Open Applications Group Middleware API Specification (OAMAS)

The OAGIS is focused on the analysis of business processes via the concept of integration scenarios. This specification provides the concepts associated with categorizing business components, the associated integration scenarios between business components, and the grouping and format of the business data, which is transferred between the components (the Business Object Document).

To address the need for a common mechanism to transmit the data between business components, independent of the format of the data, the OAG specified the Open Applications Group Middleware API Specification (OAMAS). This specification, is an attempt to have Middleware manufacturers develop their APIs in consistent format.

Additional information about the OAG, as well as the OAGIS and OAMAS may be obtained from the OAG web site.

For more detailed information the GCSS-AF Application Framework Developer's Guide provides a section on the OAGIS, and the Guide to Developing with the GCSS-AF Integration Framework provides a section on the OAMAS

5.2 Componentization

When reading the available literature today and asking prominent authors in the field, it becomes apparent that many definitions for components exists today. Therefore, it is important to define what a component is within the GCSS-AF paradigm and to delineate the properties of components. The component definition used is an adaptation of the definition provided in Component Software *Beyond Object-Oriented Programming.* Within this document the terms component and software component are used interchangeably.

GCSS-AF defines a software component as 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 useable 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.

The Guide to Developing with the GCSS-AF Integration Framework discusses components in greater detail. Topics such as: the major factors in determining the size or granularity of a component, component properties/attributes, the Component-Based Development Process and associated phases, and approaches to Wrappering and the development of Legacy Interface Components, are addressed.

6. The Future of the GCSS-AF Reference Architecture

6.1 Evolutionary Approach

Based on the iterative nature of the GCSS-AF development effort, the architecture must be able to readily support the continual integration of new capabilities and allow for a gradual migration of systems from their current legacy state over to the GCSS-AF Reference Architecture Framework. The architecture must provide for tactical as well as strategic modernization efforts due to the large legacy base of systems already in existence and continually being modernized in parallel with the GCSS-AF efforts.

Enhancement of the GCSS-AF Reference Architecture will continue, driven by the user needs, technology infusion, and the commercial information technology market.

Similarly to how the evolution of the Reference Architecture occurs over time, so too will the human interface of the system evolve. It is envisioned that GCSS-AF Human Computer Interface will be via a User's Web home page providing all the GCSS-AF capabilities necessary for that individual to complete his or her job. Access will be controlled through security services based on the definition of roles (and other access information about the user). These services will be activated at LOGIN and thus provide authorized User access to appropriate information and functionality. The definition of User's needs captured in the Operational Architecture is critical to the definition of user accessibility and roles.

Modernization of existing GCSS-AF Automated Information Systems (AIS) legacy applications will continue in parallel and drive the evolution of the GCSS-AF Reference Architecture Framework.

The selection and/or ordering of these modernization efforts for a given system depends on a variety of factors that assess the current state of the Mission Application. These factors include criteria such as:

- Current infrastructure and its expected vitality
- Adequacy of the current functionality to meet mission operational and performance requirements
- Affordability of additional modernization
- Degree to which the Mission Application has been involved in Business Process Reengineering activities
- Existence or nonexistence of as-is and/or to-be activity, process, or information flow models
- Number of interfaces to be eliminated
- Degree to which a mission application might provide significant reusable functionality within GCSS-AF.

6.1.1 Transition Phases

An important aspect of the evolution of the GCSS-AF Reference Architecture is the transition from the current implementation to the target environment over time. Table 6-1 GCSS-AF Migration provides a summary of this evolution over time for several major elements of the architecture.

Table 6-1 GCSS-AF Migration

Element	Current	Transition	Target
Applications	**Modernized Stovepipes**	**Wrapped Applications and Business Objects**	**Business Objects**
Mission Application Interfaces	**Language Dependent Point to Point API**	**Wrapped API w/ BSR Interface**	**Business Service Request**
Transaction Processing	**File Transfers and Disk Sharing**	**COTS MOM**	**Direct MOM support through ORB**
Legacy Interfaces	**Point to Point**	**Single Legacy Module Wrapped w/ ORB**	**Legacy Interface Component**
Enterprise Data	**Relational**	**Relational Wrapped w/ ORB Services**	**Fine-grained Business Components**
Desktop	**OLE, DCOM**	**COM+ w/ Bridge to ORB**	**Office Components**

This Page Left Intentionally Blank.