Application of Middleware in Developing the Resource Management Segment (RMS) of the AFSCN

Ron Toms - Lockheed Martin Federal Systems
(ron.toms@lmco.com)
Bill Glocker - Hughes Information Technology Systems
(bill.glocker@lmco.com)
Overview

- RMS Design Goals
- RMS Software Infrastructure Design Goals
- RMS Software Infrastructure Architecture
- RMS Software Infrastructure Key Points
- Summary/Work in Progress
RMS Design Goals

- Utilize COTS for core components which have a well-defined and understood function.
  - Sybase SQL Server as RDBMS
  - Sun Solaris 2.5.1 as the operating system
  - Qualix HA+ for High Availability
  - Sun SunNet Manager for Network Management
- Utilize toolkits as a means of constructing applications for which no clear-cut COTS package meets the requirements of the AFSCN.
  - Delphi class libraries from Hughes Aircraft for Scheduling
  - UIM/X GUI builder for creating displays
  - RogueWave class libraries for general purpose C++ class libraries and database access
- Develop a middleware based infrastructure (CORBA 2.0)
RMS Software Infrastructure Design Goals

- Utilize CORBA 2.0 based products
  - Selected Iona Technologies Orbix 2.0 as base CORBA product
- Separate applications from core services
  - Applications do not talk directly to each other
  - Applications have no concept of databases; Only concerned with data
  - GUIs are separate from application engines
- Control data flow through well-defined Interface Definition Language (IDL) interfaces
- Manage object persistence
- Manage client concurrency
- Insulate application code from infrastructure implementation
RMS Software Infrastructure Architecture Key Points

• Interface classes/Server Wrapper - define the interfaces to other applications.
  – CORBA communication between clients
  – Insulates applications from CORBA implementation specifics (i.e. applications only have to handle exceptions defined in the IDL, not all of the standard CORBA exceptions)

• Persistence Agents
  – Objects will exist in “address space”
  – Reduces overhead caused by CORBA transaction and network communication
  – Uses RogueWave class libraries (dbtools.h) to access/update database directly
  – Allows the manipulation of “object” all at once

• Supports all elemental CORBA 2.0 types
RMS Software Infrastructure Architecture Key Points

• Streaming objects for CORBA communication
  – Uses unbounded sequence of characters
  – Objects know how to stream themselves (write objects to/from ASCII buffers via << and >> operators)
  – Supports both Ada and C++ implementations

• Data Distributor - handles write-locks, concurrency, subscriptions, and update notifications
RMS Software Infrastructure Status

- Have begun actual implementation of the architecture
- Work in progress
  - Failure detection tools
  - Detailed Performance analysis
  - Evaluating multi-threaded CORBA (Orbix-MT)