Applying A Distributed Architecture to Real Time Hardware Control and Status

Rob Andzik
andzik@rtlogic.com

RT Logic!
1042 Elkton Drive
Colorado Springs, CO 80907
Phone: 719-598-2801
FAX: 719-598-2655
www.rtlogic.com
Presentation Overview

- Introduction
- Motivation
- Ground system software architecture
- Scalability
- Performance
- Summary
Introduction

• RT Logic
  – Satellite Ground Systems
  – Satellite Test Systems
  – Launch and Range Systems

• Wide variety of real time systems and applications
Motivation For A Distributed Architecture

External Factors
- Desire for COTS products to increase “in box” functionality while decreasing cost
- Industry/Government standards are gaining momentum
  - DII/COE, JTA, TENA
  - CCSDS, CORBA, TCP/IP, SNMP, XML
- Increased number of Multi-Mission systems
- Heterogeneous computing environments
- Multiple programming languages
- Eliminate Proprietary Interfaces

Internal Factors
- Significant advancements in hardware technologies
- Improve performance and quality of service
- Support both backward and forward compatibility
- Decrease the time required in all aspects of a project: proposal to delivery
- Extend system capabilities beyond the embedded system
Defining The Architecture

Key Observations
- It is imperative that a software architecture be defined at an appropriate level of abstraction for it to remain a viable solution over time.
- Interface definitions and constancy are key to the stability of a software architecture.
- The inability of an architecture to accommodate unexpected fluctuation of both requirements and interfaces often results in the *software death spiral*.

Lessons Learned From Past Efforts
- The object oriented taxonomy of a system must take into account the dynamics and evolution of that system over time.
- The first definition of an interface, no matter how well thought out, is often incorrect.
- Latent hidden requirements often force subtle modifications to interfaces that over time create complex and costly interdependencies between modules.
Telemetrix Architecture

- Middleware
  - CORBA
    - Platform and language independence
- Components
  - Based on the Extension Interface design pattern.
    - *Pattern-Oriented Software Architecture, VII.* Douglas Schmidt et al.
  - Interfaces are aggregated and independent of the component.
  - Interfaces and functionality may evolve without destabilizing the system.
  - A designer may use interface adaptors and decorators to add arbitrary functionality.
- Services
  - Combine components into a common set of functionality designed to meet a focused set of requirements.
    - Resource Management, Configuration Persistence, Archive/Playback, Decommutation, Logging, Event Distribution, FD/FI as well as standard CORBA services.
- Framework
  - Combine existing services into a cohesive framework upon which a variety of systems may be developed.
- Telemetrix System
  - Integrated distributed system combining various aspects of the framework
• VxWorks Chassis
• Local services used for managing device components
• Applications on workstations utilize services remotely
• Device components control hardware functionality
• Generic IDL interfaces simplify remote control & status
• Components can simultaneously support multiple interfaces
Remote Sites

- Services distributed to local workstation.
- Automated system configuration using Resource Manager.
- Automated pre-pass verification using FD/FI
- Common configurations between all sites.

Local Workstation

- Resource Manager
- Site Allocator
- Configuration Service
- Logging Service
- Event Service
- Naming Service

TCP/IP (Interfaces Defined Using CORBA IDL)

RF Chassis

Baseband Chassis

Comm Chassis

SOC

RT Logic! Enabling Your Solutions
Control Centers

- Resource Manager handles multiple contacts simultaneously
- Seamlessly integrate TT&C components into framework
- Control of remote sites possible
- Hierarchical resource management
• CORBA Performance Tests
  – Using the TAO ORB implementation.
  – Communication between 2 Intel Pentium III ™ workstations over a 100 Mbit network.

• Single CORBA call
  – Oneway ping: 0.2 - 0.3 msec
  – Twoway ping: 0.8 - 1.1 msec

• Throughput
  – 50Mbits/sec possible with large packet sizes
Event Distribution

- Event Service
  - Same test platforms
  - Varied size of the event payload
  - Measured maximum throughput and number of event per second
- NOTE: Bundling frames of telemetry improves throughput
Summary

- Components
  - Support backward compatibility
  - Permits adaptation of additional functionality and interfaces.
    - Including non-CORBA middleware such as SNMP etc.
- Software Framework
  - Permits rapid development of complex systems
  - Services/Components may be modified as a system evolves without adversely effecting other parts of the system
- Distributed Open Architecture
  - Components and services may be relocated to accommodate system design and performance requirements.
  - Easily scales to handle multi-mission satellite operations.
  - Breaks the traditional proprietary COTS system approach