Carnegie Mellon
Software Engineering Institute
Pittsburgh, PA 15213-3890

Architectural Aspects of Long-Lived Ground Systems

Charles ("Bud") Hammons

Ground Systems Architecture Workshop 2006

Sponsored by the U.S. Department of Defense
© 2006 by Carnegie Mellon University

Report Documentation Page

Form Approved
OMB No. 0704-0188

Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
2006		**00-00-2006 to 00-00-2006**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
Architecture Aspects of Long-Lived Ground Systems	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **Carnegie Mellon University,Software Engineering Institute,5000 Forbes Avenue,Pittsburgh,PA,15213-3890**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**11**	

Carnegie Mellon
Software Engineering Institute

Topics

Ground Systems Challenges

A motivating example - TSAT*

Architecture Strategy

Architecture Tactics

Realization

Summary / Q&A

What architecture can do to support system longevity

***Disclaimer: personal views, not necessarily those of the Transformational Satellite Communications (TSAT) PMO**

Carnegie Mellon
Software Engineering Institute

Ground Systems Challenges

- Unprecedented Operational Capability
- Interoperability with external systems also in development
- Interoperability with Legacy Systems
- Evolution in CONOPS
- Evolution in protocols and underlying technology

- Architecturally significant attributes
- Drive lifecycle evolution/change into development cycle

Carnegie Mellon
Software Engineering Institute

A Motivating Example - TSAT

Goals include

- mission-critical satellite-based packet and circuit communications for the warfighter
- quality of service, info assurance, comm. on the move,…
- seamless integration into the Global Information Grid (GIG)
- complex interactions with military planners/systems

Other programs have similarly challenging objectives and complexity (e.g. business enterprise integration exploiting RFID*, network communications,…)

Overarching Challenge – develop a large, complex, long-lived, software intensive systems in an environment that is fluid both during and after development

*RFID – Radio Frequency IDentification

Carnegie Mellon
Software Engineering Institute

Architecture Strategy

At the risk of stating the obvious, identify what is fixed, what is variable

Fixed/Slow-moving
- domain-specific data
- *essential* behavior
- software/hardware split

Variable/Evolving
- standards, protocols
- external interfaces
- CONOPS, deployment
- time constraints
- value-added features
- technology refresh
- human-machine task split

Tactics: identify architectural features that allow change and protect invariants

Carnegie Mellon
Software Engineering Institute

Architecture – Tactics 1

Separation of Concerns

Explicit domain-specific data model

- most resilient piece of large system-of-systems
- desirable to version elements
- unambiguous units of measure
- include behavior with roles, permissions, etc.

Separate CONOPS from data model

- CONOPS is mechanized as an explicit element of architecture
- captures policies that drive behavior
- describes human-machine task division

Separate domain-specific behavior from supporting infrastructure

Carnegie Mellon
Software Engineering Institute

Architecture – Tactics 2

Define Capable Infrastructure

Generalized inter-component communications
- messaging 'middleware'
- asynchronous to near real-time constraints
  - multiple transport mechanisms transparent to application components

Explicit management model for components
- formal model for control and monitoring
- 'component registry'
- include version as lookup criteria
- enable automated & remote component

Isolate external interfaces from applications/services

Carnegie Mellon
Software Engineering Institute

Architecture – Tactics 3

Exploit Legacy & COTS Software

- Treated as components in architectural model
- Individual choices should neither “break” nor drive architecture
- Unique structure hidden by common packaging conventions
- On case-by-case basis, revision/replacement is pre-planned

Carnegie Mellon
Software Engineering Institute

Realization 1

Architectural Styles

- Client-Server
- Service-Oriented Architecture (SOA)
- Agent-based systems
- Hybrids

Communications Models

- XML-based (including “Web Services”)
- CORBA and relatives
- Problem-specific binary communications protocols (e.g. WSTAWG* real-time model)

*WSTAWG – Weapons System
Technical Architecture Working Group

Carnegie Mellon
Software Engineering Institute

Realization 2

Organizational Issues

- Recognize going in that this is difficult work
- Requires organizational buy-in and sustained management attention
- Expect numerous objections
- Complexity and long time frame ensures mistakes will happen – architecture can mitigate effects when domain mutates or market forces influence what is available or appropriate

Carnegie Mellon
Software Engineering Institute

Summary

• Developing complex net-centric systems while we are still trying to fully understand what it means to be net-centric represents unique opportunities and risks

• Rapid evolution in technology, standards, and protocols increases variability that programs must comprehend.

• Architecture can mitigate some of the difficulties.

• There is *still* no silver bullet.