Software Architecture for Satellite Ground System Development

Jeff Garland

Principal Consultant

CrystalClear Software, Inc

Nature of Ground Systems

- Highly Complex, Large-Scale, Mission Critical
- Distributed / Concurrent systems
- Many COTS components -- require integration
- Heterogeneous components
  - Multiple languages
  - Multiple storage techniques etc

Key Requirements of the Approach

- Need an approach that scales to huge systems
- Need to be understandable by large range of project participants
  - minimal use of modeling elements (eg: complex UML)
- Need to map to 'real things'
- Needs a focus on 'interfaces'
- Needs to include quantitative and other annotations

Modeling Software Architectures

- UML -- Unified Modeling Languages
- IEEE 1471 -- Views and Viewpoints
  - Viewpoint -- template for a view (purpose, applicability, stakeholders)
  - View -- instance of a viewpoint
- 14 Viewpoints in all
- Key Elements to Model
  - Components and Interfaces
  - Subsystems
  - Processes and Databases

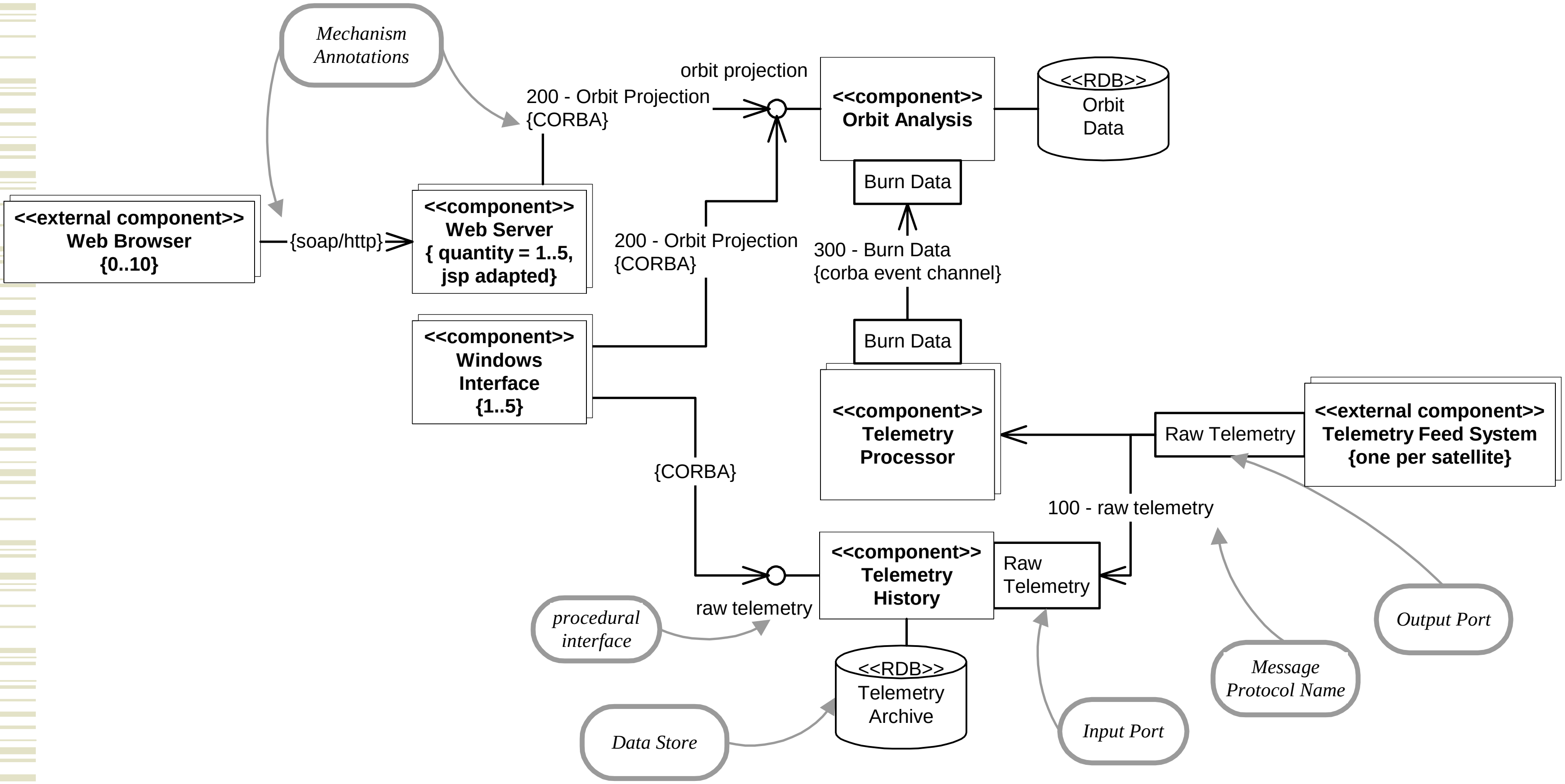
Top 5 Views

Viewpoint	**UML Diagram Type**	**Description**
Context	Use Case	Show the external system actors and the system under design.
Component	Component	Illustrate component communications.
Component Interaction	Interaction	Interactions among components.
Layered Subsystem	Package	Illustrate layering and subsystems design.
Deployment	Deployment	Mapping of software to hardware for distributed systems.

Component Viewpoint

Purpose	Describe runtime component connectivity and communication. Can be applied to performance analysis and later the process interaction design.
When Applicable	During system design and development, as analysis views and subsystems are identified.
Stakeholders	Architecture Team, Subsystem Developers, Test Team, Software System Engineering Team, Systems Engineering Team, Project and Development Managers (to a lesser degree)
Scalability	Drawn with scenario or component focus, Can make use of composite components.
Relation to Other Views	The Component Views should be consistent with components shown in the Process and Deployment Views.

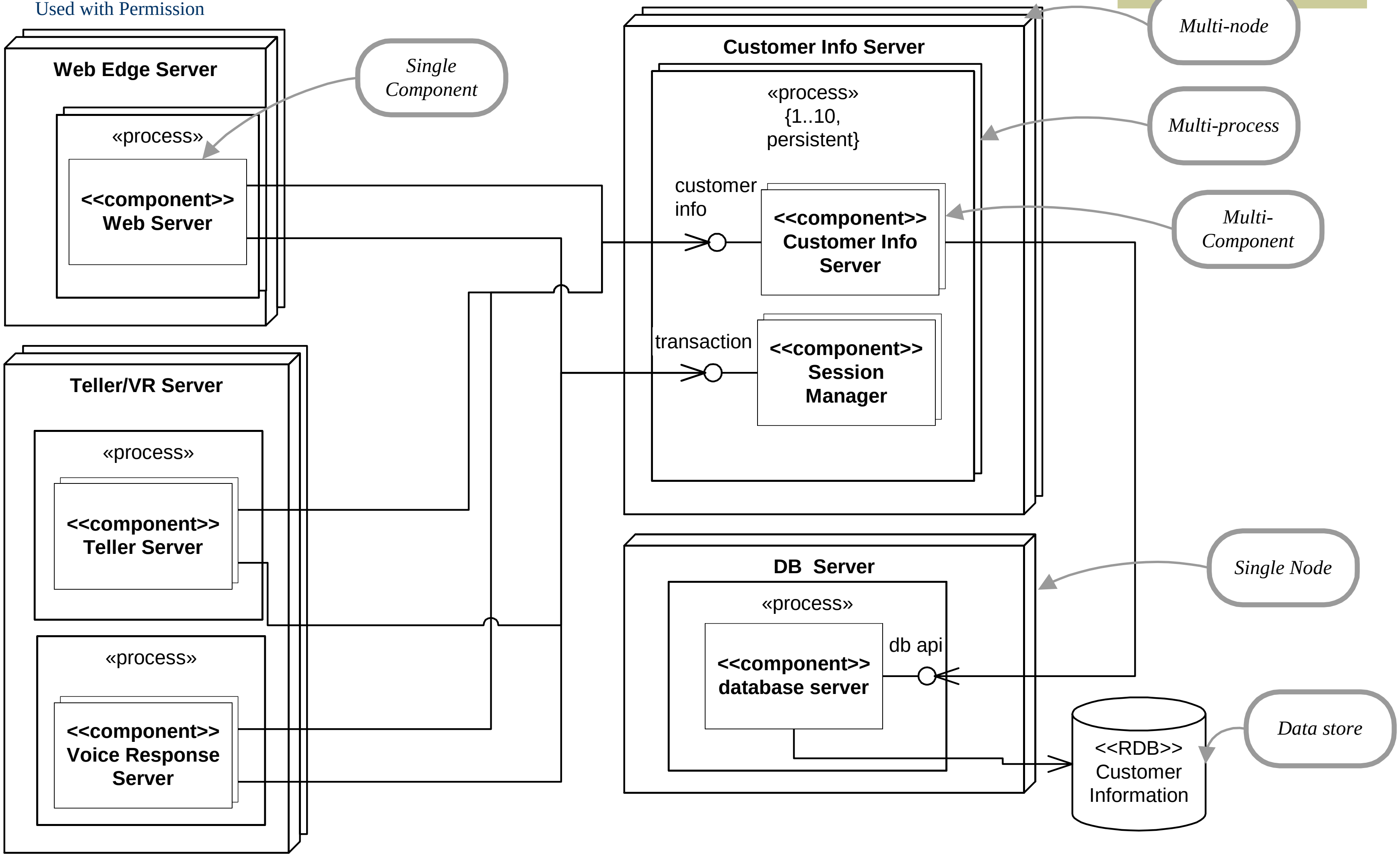
Component View Example


Mechanism Annotations
orbit projection
200 - Orbit Projection {CORBA}
<<component>> Orbit Analysis
<<RDB>> Orbit Data
Burn Data
<<external component>> Web Browser {0..10}
{soap/http}
<<component>> Web Server { quantity = 1..5, jsp adapted}
200 - Orbit Projection {CORBA}
300 - Burn Data {corba event channel}
<<component>> Windows Interface {1..5}
Burn Data
<<component>> Telemetry Processor
Raw Telemetry
<<external component>> Telemetry Feed System {one per satellite}
{CORBA}
100 - raw telemetry
<<component>> Telemetry History
Raw Telemetry
procedural interface
raw telemetry
<<RDB>> Telemetry Archive
Data Store
Input Port
Message Protocol Name
Output Port

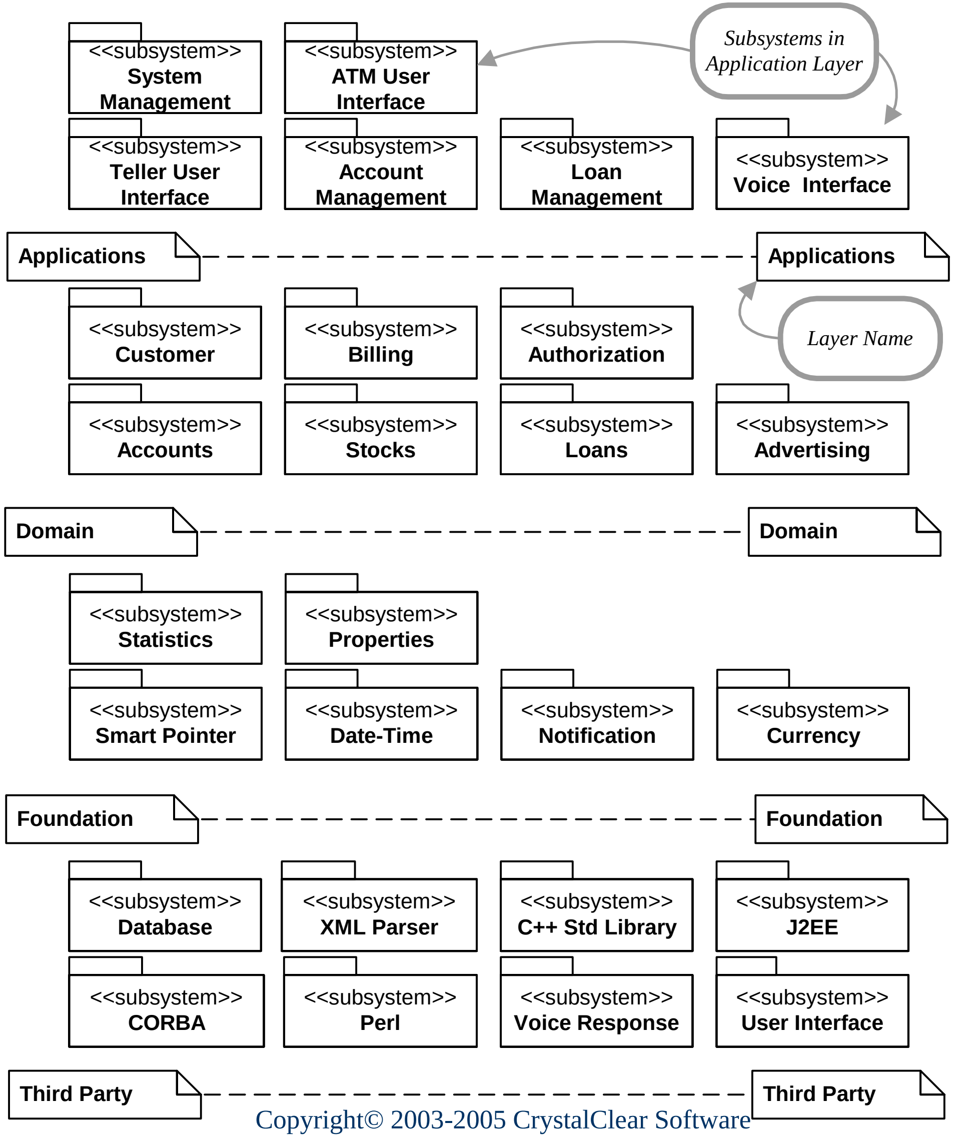
Component Description Table Example

Component Name	Role
Teller Client	Provide a user interface tuned for the needs of bank tellers.
Teller Server	Provides services for the use of tellers. This includes administrative functions.
Session Manager	Provides transaction and session id support.
Customer Info Server	Provides service interfaces that provide access to basic customer information.
…	

Deployment View Example



Layered Subsystem View

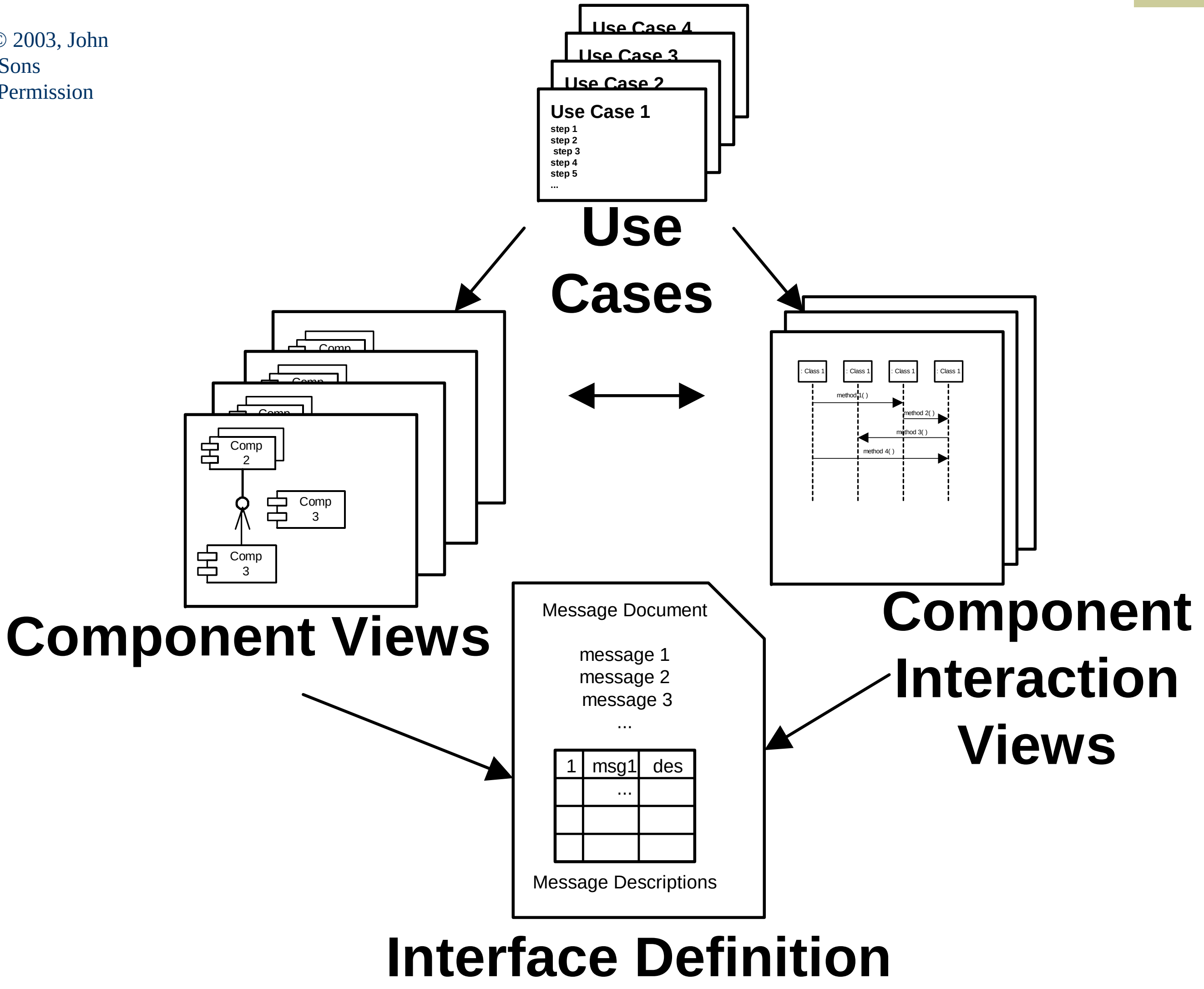


Subsystems in Application Layer
<<subsystem>> System Management
<<subsystem>> ATM User Interface
<<subsystem>> Teller User Interface
<<subsystem>> Account Management
<<subsystem>> Loan Management
<<subsystem>> Voice Interface
Applications
Applications
Layer Name
<<subsystem>> Customer
<<subsystem>> Billing
<<subsystem>> Authorization
<<subsystem>> Accounts
<<subsystem>> Stocks
<<subsystem>> Loans
<<subsystem>> Advertising
Domain
Domain
<<subsystem>> Statistics
<<subsystem>> Properties
<<subsystem>> Smart Pointer
<<subsystem>> Date-Time
<<subsystem>> Notification
<<subsystem>> Currency
Foundation
Foundation
<<subsystem>> Database
<<subsystem>> XML Parser
<<subsystem>> C++ Std Library
<<subsystem>> J2EE
<<subsystem>> CORBA
<<subsystem>> Perl
<<subsystem>> Voice Response
<<subsystem>> User Interface
Third Party
Third Party

Where Has this Approach Been Applied?

- Genesis was development of Iridium Ground-System
  - Was pre-UML at the time
  - Refined on projects since then
- Framework documentation for another satellite system
- Other large distributed systems
  - financial, communications, enterprise systems
- Other satellite ground systems

Systematic Elaboration of Architecture



Use Cases

Component Views

Component Interaction Views

Interface Definition

Evolution

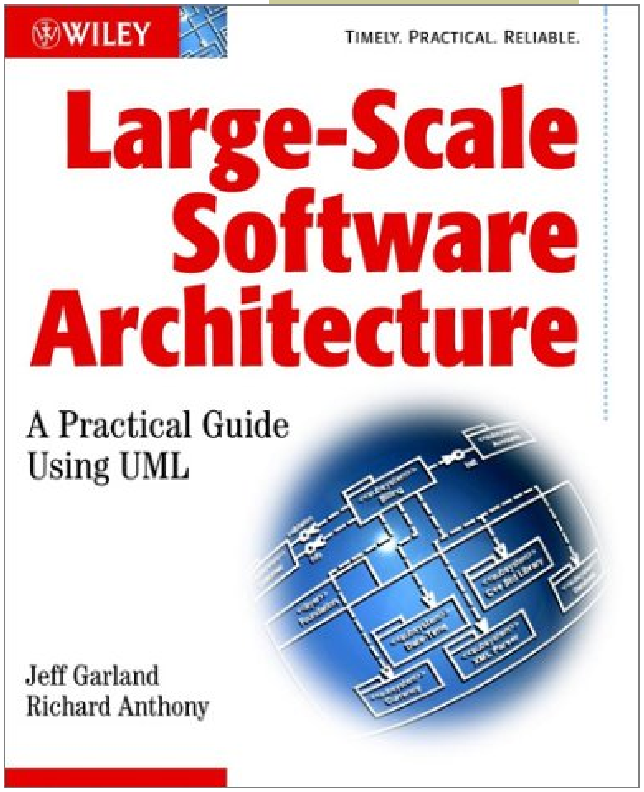
- Runtime
  - Understand the runtime evolution requirements and constraints
- Buildtime
  - Subsystem dependencies is key
- Be Aware of Hard to Evolve Parts of Architecture
  - Data == concrete
  - COTS difficult to upgrade

Evaluation

- Evaluation Issues
  - Complexity of systems is a barrier
  - ‘architectural view’ doesn’t reflect implementation
- Recommendations
  - Focus on key scenarios
  - Use appropriate views to focus
  - Stay as concrete as possible
  - Get an experienced software architect

Conclusions and More Info

- Systematic development of software architecture is achievable
- Tools are a problem
- UML issues


WILEY
TIMELY. PRACTICAL. RELIABLE.
Large-Scale Software Architecture
A Practical Guide Using UML
Jeff Garland
Richard Anthony

- More info at www.largescalesoftwarearchitecture.com
- Email: jeff@crystalclearsoftware.com