Architecting Large-Scale Systems

Peter Eeles
Executive IT Architect, IBM
peter.eeles@uk.ibm.com
Agenda

What is a large-scale system?
- The importance of architecture
- Large-scale architecture in practice
- A worked example
- Summary
Rational Unified Process
Large-scale initiatives

- Large-scale initiatives extend beyond a single software development project
  - Single
  - Software
  - Development
  - Project
    - Multiple?
    - Software / hardware / people / information?
    - Development / operations?
    - Programme?
Large-scale initiatives

- Enterprise architecture
  - Defining an architecture that underpins a number of systems

- Strategic reuse
  - Developing reusable assets that are used within a number of systems

- Systems engineering
  - Developing a system that contains elements of hardware, software, workers and data

- Enterprise Application Integration
  - Developing a solution that includes the integration of a number of legacy systems

- Packaged application development
  - Developing a solution that includes the configuration of a packaged application, such as an ERP or CRM solution

- Outsourced development
  - Defining an architecture that lends itself to the outsourced development of its constituent parts, whilst ensuring the quality and integrity of these parts

- Service-Oriented Architecture
  - Supporting the creating of composite applications whose parts are reusable services

Architecture is key!
Agenda

- What is a large-scale system?

The importance of architecture
  - Large-scale architecture in practice
  - A worked example

Summary
Various architecture-related papers available at …

- www.architecting.co.uk
### Architecture

- **Architecture** is the fundamental **organization** of a **system** embodied in its **components**, their **relationships** to each other, and to the **environment**, and the **principles** guiding its design and evolution. [IEEE 1471]

- The software architecture of a program or computing system is the **structure** or structures of the system, which comprise software **elements**, the externally visible properties of those elements, and the **relationships** among them. [Bass]

- [Architecture is] the organizational **structure** and associated **behavior** of a system. An architecture can be **recursively decomposed** into **parts** that interact through interfaces, **relationships** that connect parts, and **constraints** for assembling parts. Parts that interact through interfaces include classes, components and subsystems. [UML 1.5]
Architecture

- An architecture defines structure
- An architecture defines behaviour
- An architecture is concerned with significant elements
- An architecture meets stakeholder needs
- An architecture conforms to an architectural style
- An architecture is influenced by its environment
- An architecture influences organizational structure
- An architecture is present in every system
- An architecture embodies decisions based on rationale

“The life of a software architect is a long and rapid succession of suboptimal design decisions taken partly in the dark.” [Kruchten]
An architecture comes in many forms

- Enterprise Architecture
  - System Architecture
    - Software Architecture
      - Application Architecture
    - Technical Architecture
    - Hardware Architecture
  - Organization Architecture
The benefits of architecting

- Architecting helps manage complexity
- Architecting ensures architectural integrity
- Architecting provides a basis for reuse
- Architecting addresses system qualities
- Architecting drives consensus
- Architecting reduces maintenance costs
- Architecting supports impact analysis
- Architecting supports the planning process
Agenda

- What is a large-scale system?
- The importance of architecture

Large-scale architecture in practice
- A worked example
- Summary
Enterprise, Business, System

- **Enterprise**
  - Set of resources that are used to meet a business need or mission
  - Enterprises can cross organization and even business boundaries
  - Enterprises provide value to their stakeholders (e.g. stockholders, community, nation, etc.)

- **Business (Organization)**
  - A part of an enterprise responsible for one or more business processes (may also be Business Unit, Segment, etc.)

- **System**
  - An entity consisting of hardware, software, workers and information ... that provides services used by an enterprise in meeting its purpose or mission
A System

- ... is made up of
  - Software
  - Hardware
  - Workers (people)
  - Information (data)
A System of Systems

- Consider a system to be made up of a collection of other systems, each made up of software, hardware, workers and information
  - A “system of systems”
An Example

Level 1

- NGST System
  - Plan for NGST System
    - Inception
    - Elaboration
    - Construction
    - Transition

Level 2

- Observatory
  - Optical Telescope Element (OTE)
  - Spacecraft
  - Integrated Science Instrument Module (ISIM)
    - Instruments
    - Infrastructure
  - Plan for Observatory
    - Inception
    - Elaboration
    - Construction
    - Transition

Level 3

- Ground Segment
  - Plan for Ground Segment
    - Science Operations Center (SOC)
    - Communications Element

- Launch Segment
  - Plan for Launch Segment
    - Launch Vehicle
    - Payload Adapter
    - Launch Site Services
Architectural Representation

- IEEE-1471
  - The IEEE Recommended Practice for Architectural Description of Software-Intensive Systems
  - This standard provides a conceptual framework for architectural description and defines what is meant by a 1471-compliant architectural description

- 4 + 1 Views of Software Architecture
- Siemens
- DoDAF
- MoDAF
- ToGAF
- RM-ODP
- The Zachman Framework
- RUP for Systems Engineering (RUP-SE)
Describing an Architecture – Kruchten 4+1 views
Describing an Architecture – Cantor (RUP-SE)

<table>
<thead>
<tr>
<th>Viewpoint Level</th>
<th>Worker</th>
<th>Logical</th>
<th>Information</th>
<th>Physical</th>
<th>Process</th>
</tr>
</thead>
<tbody>
<tr>
<td>Context</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Analysis</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Design</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Implementation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Agenda

- What is a large-scale system?
- The importance of architecture
- Large-scale architecture in practice
- A worked example
- Summary
An example

- A retail store
- Selling books, videos, DVDs, music CDs, etc.
Is a sales clerk inside or outside the system?
Level 1: Context Diagram (initial)
Level 1: Use-Case Model

- Customer
  - Request Information
- Auditor
  - Audit Business
- Sponsor
  - Review Business Case
- Order Goods
Level 1: Use-Case Model

- Basic Flow of the “Order Goods” Business Use Case
  - The use case starts when the Customer initiates the placing of an Order for Products.
  - An appropriate Order is placed that contains the Products to be purchased, along with the relevant quantity of each Product. The Customer receives the ordered Products and a request for payment.
  - The Customer pays for the Order.
  - The use case ends.
Level 1: Use-Case Model

- Basic Flow of the “Order Goods” Business Use Case
- The system is treated as a “black box”
  - How the order is fulfilled and payment requested is internal to the system
Level 1: Context Diagram (partial)

Customer

Auditor

Sponsor

«System»

Retail System

Place order (details: Order Details)

Make payment (order: Order, payment details: Payment Details)
Level 1: Operation Realization

- For “Place order” operation
- The system is treated as a “white box”
Level 1: Operation Realization

- For “Make Payment” operation
From Level 1 to Level 2
Level 2: Context Diagram

Sales Clerk

Accounting Clerk

- Send invoice
- Process Payment
- Perform credit check
Level 1: Operation Realization

- What about non-functional requirements?
- What about other viewpoints (other than logical or worker)?
## Describing an Architecture – Cantor (RUP-SE)

<table>
<thead>
<tr>
<th>Viewpoint</th>
<th>Worker</th>
<th>Logical</th>
<th>Information</th>
<th>Physical</th>
<th>Process</th>
</tr>
</thead>
<tbody>
<tr>
<td>Context</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Analysis</td>
<td>Subsystem</td>
<td>Subsystem</td>
<td></td>
<td>Locality</td>
<td></td>
</tr>
<tr>
<td>Design</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Implementation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Level 1: Operation Realization

- For “Place order” operation
- This is “Joint realization” across different viewpoints (logical, worker, physical)

<table>
<thead>
<tr>
<th>Step</th>
<th>Action Performed</th>
<th>Subsystem</th>
<th>Locality</th>
<th>Budgeted Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The order details are taken</td>
<td>Sales Clerk</td>
<td>Branch</td>
<td>60 seconds</td>
</tr>
<tr>
<td>2</td>
<td>A credit check is performed</td>
<td>Accounting Clerk</td>
<td>Central Office</td>
<td>10 seconds</td>
</tr>
<tr>
<td>3</td>
<td>The products are shipped to the customer</td>
<td>Fulfilment Clerk</td>
<td>Warehouse</td>
<td>1 day</td>
</tr>
<tr>
<td>4</td>
<td>An invoice is sent to the customer</td>
<td>Accounting Clerk</td>
<td>Warehouse</td>
<td>1 day</td>
</tr>
</tbody>
</table>
Joint Realization
The “System of Interconnected Systems” Pattern

- An example using the Rational Unified Process

<table>
<thead>
<tr>
<th>Requirements</th>
<th>Analysis &amp; Design</th>
<th>Implementation</th>
<th>Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use-Case Model</td>
<td>Analysis Model</td>
<td>Implementation Model</td>
<td>Test Artifacts</td>
</tr>
<tr>
<td>Analysis Model</td>
<td>Design Model</td>
<td>Test Artifacts</td>
<td></td>
</tr>
<tr>
<td>Design Model</td>
<td>Test Artifacts</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Implementation Model</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Superordinate System

Subordinate System
Programme / Project Governance

- Programme concerns
  - Alignment of projects within a programme
Programme / Project Governance

- Alignment of project management work products
  - Programme / project vision
  - Programme / project plans (schedules, budgets, signoff points, funding, releases)

- Alignment of project management processes
  - Scope (requirements) management
  - Change management
  - Test management
  - Risk and issues management
  - Quality management
  - Measurement / metrics gathering
  - Programme / project management reviews
  - Configuration management
  - …
Architectural (Solution) Governance

- Architectural concerns
  - Alignment of subordinate systems with the superordinate system

- Alignment of architectural work products
  - Requirements model
  - Design model
  - Implementation model
  - Data model
  - Standards and guidelines
  - Infrastructure definition

- Alignment of architectural processes
  - Identification / refinement of interfaces and components
  - Identification / refinement of architectural properties (cost, performance)
  - Architecture reviews
  - …
Summary

- “Systems” thinking requires us to think beyond software
  - Systems engineering, enterprise architecture, strategic reuse, …
- Certain qualities cannot be achieved by software alone
  - Performance, reliability, …
- Software/systems engineering principles and practices can scale to support the development of large-scale systems
- The “system of interconnected systems” pattern provides a means of managing complexity within such initiatives
Additional Resources

- What is a Software Architecture?

- Characteristics of a Software Architect

- The Process of Software Architecting

- The Benefits of Software Architecting

- Hardware/software codevelopment using a model-driven systems development (MDSD) approach
Questions
Thank You

Peter Eeles
peter.eeles@uk.ibm.com