

Architecting Large-Scale Systems

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

WebSphere User Group

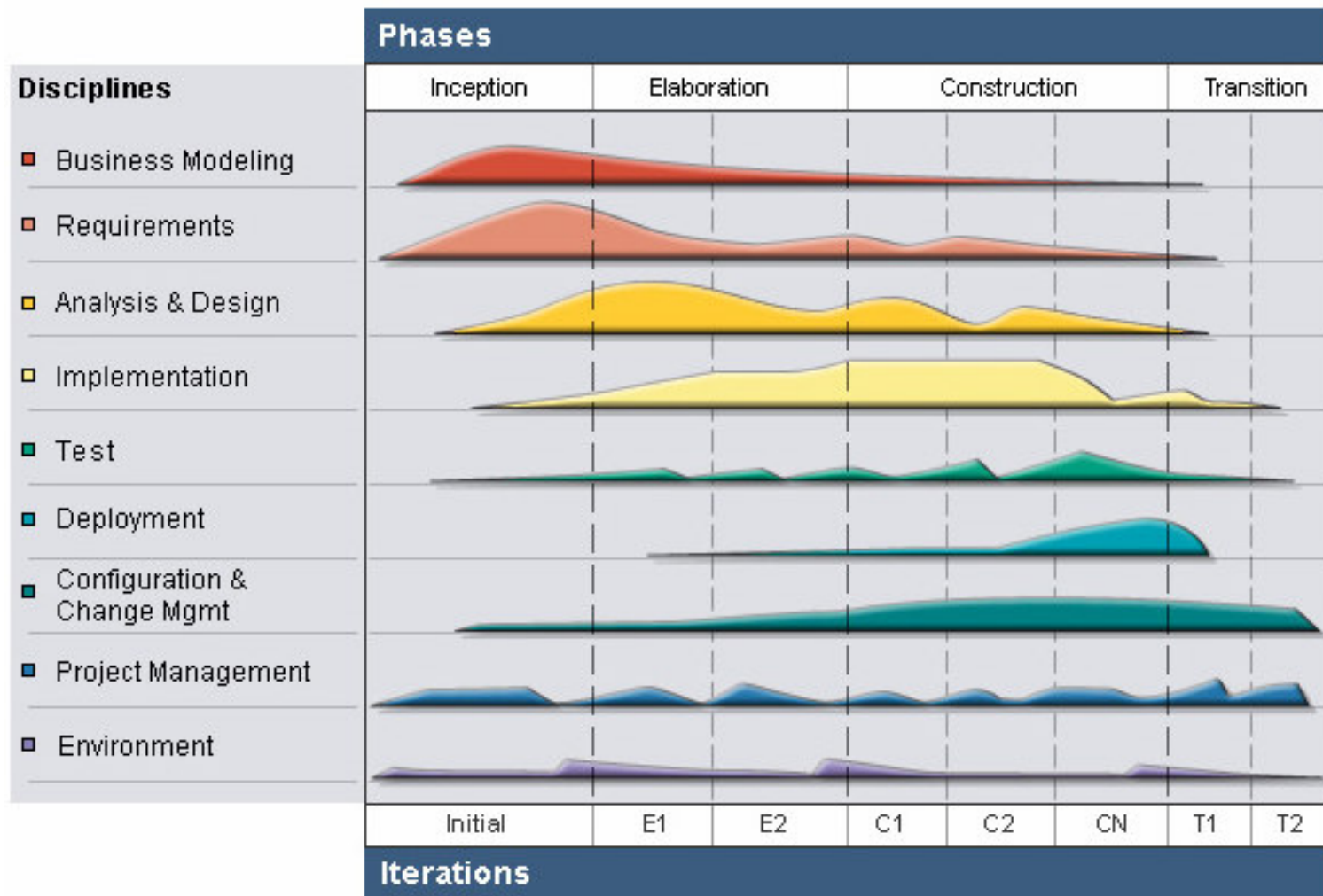


17 September – Edinburgh

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 Multiple?
 - ▶ Software Software / hardware / people / information?
 - ▶ Development Development / operations?
 - ▶ Project 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

The screenshot shows the homepage of the 'Software Architecting' website. The title 'Software Architecting' is in a large, green, stylized font. Below it is the URL 'www.architecting.co.uk'. A green navigation bar contains links: Home | Contact | Books | Links | Papers | Presentations | Workshops. The main content area has a welcome message from Peter Eeles, stating the site is dedicated to software architecture and provides resources. It also includes a brief bio and a note about the site being under construction. A diagram illustrates the relationships between 'Architect', 'Architecting', and 'Architecture'. 'Architect' performs 'Architecting' and is responsible for 'Architecture'. 'Architecting' modifies 'Architecture'.

Software Architecting
www.architecting.co.uk

Home | Contact | Books | Links | Papers | Presentations | Workshops

Welcome to Peter Eeles' Software Architecting website! This site is dedicated to the process of architecting software systems and provides access to various resources (papers, presentations) that I've created.

In brief, I'm an Executive IT Architect within IBM's Rational Software Services organisation. [Click here](#) for a (very) brief bio. This site is under construction so please call back soon.

```
graph TD; Architect[Architect] -- performs --> Architecting[Architecting]; Architect -- responsible for --> Architecture[Architecture]; Architecting -- modifies --> Architecture;
```

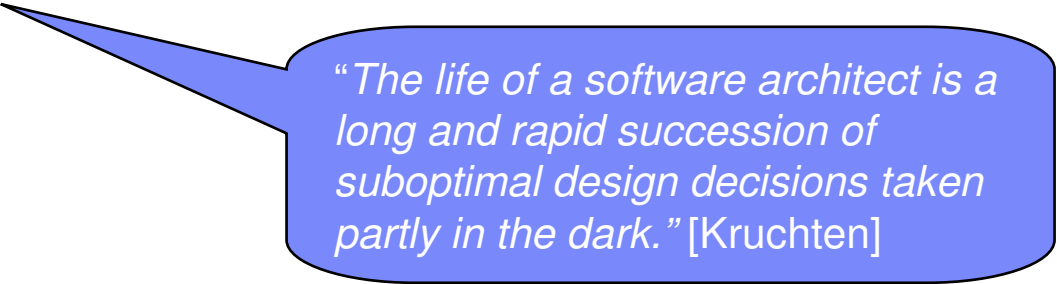
Copyright © Peter Eeles 2002-2007
All rights are reserved

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]*

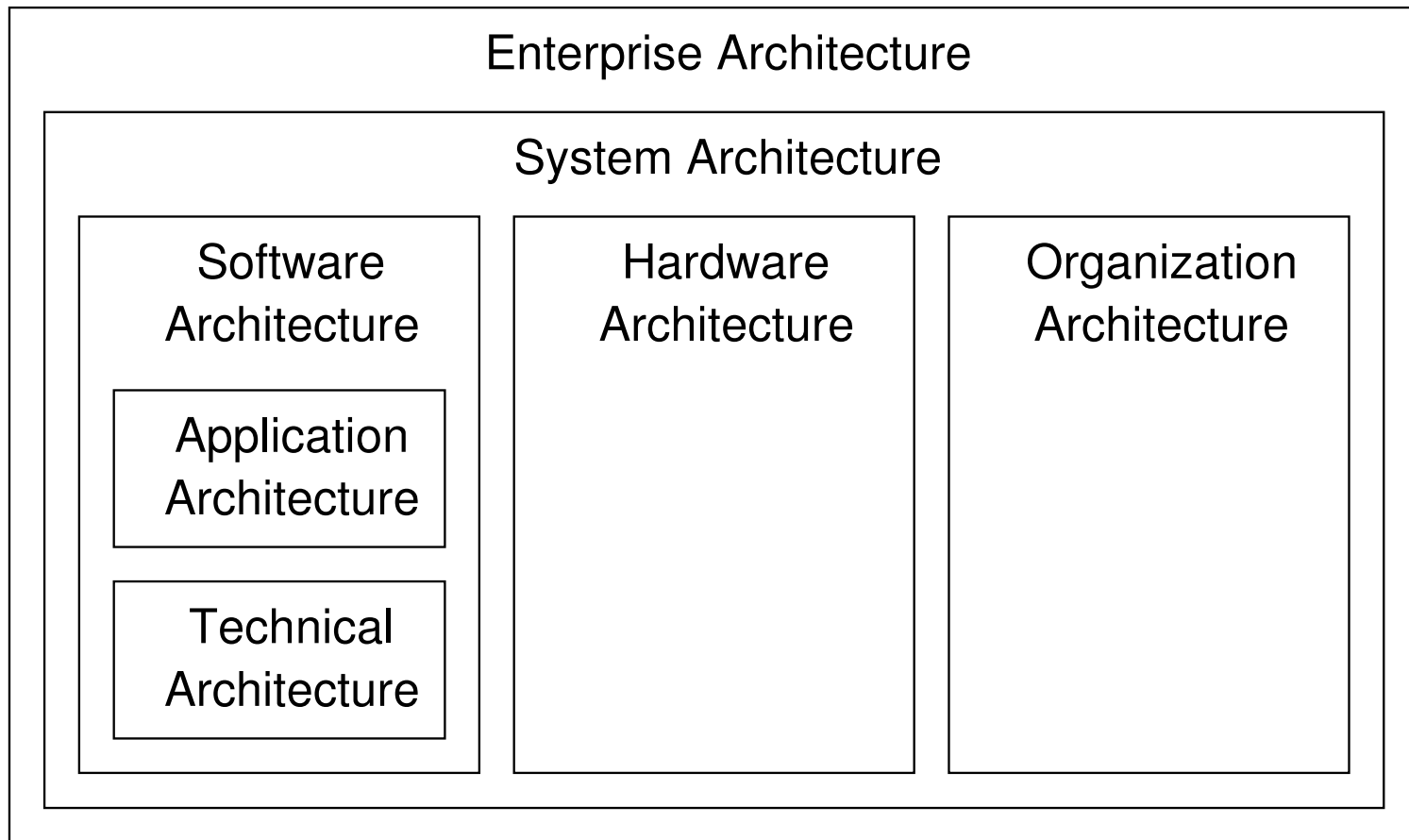
Architecture

- 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



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

```
*** STOP: 0x0000001E (0x80000003,0x80102090,0x00000000,0xFF68CCAC)
Unhandled Kernel exception 80000003 from 80102090 (0, ff68ccac).
*** Address 80102090 has base at 80100000 - ntoskrnl.exe
```

```
eax=ffdf13c ebx=ff68cf60 ecx=ff68ce2c edx=8016484e esi=00000000 edi=8019e3d0
eip=8014fbc2 esp=ff68cb5c ebp=ff68cf7c p4=0300      nv up di ng nz na po nc
cr0=80050039 cr2=8017ddd0 cr3=00030000 cr4=00000000 iopl:1f      efl=ff68cb54
gdtr=80036000  gdtl=03ff idtr=80036400  idtl=07ff tr=0028  ldtr=0000
```

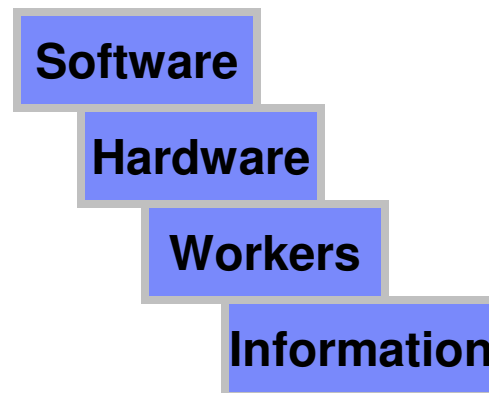
Dll Base	DateStmp	- Name	Dll Base	DateStmp	- Name
80100000	2c921d20	- ntoskrnl.exe	80400000	2c7d4b45	- hal.dll
80010000	2c360942	- Atdisk.sys	80259000	2c42f49a	- Fastfat.sys
fcc00000	2c360940	- Floppy.SYS	fcc10000	2c3609c5	- Hpfs_Rec.SYS
fcc20000	2c360952	- Null.SYS	fcc30000	2c360925	- Beep.SYS
fcc40000	2c360945	- i8042prt.SYS	fcc50000	2c36094d	- Mouclass.SYS
fcc60000	2c36094a	- Kbdclass.SYS	fcc70000	2c360901	- Videoprt.SYS
fcc80000	2c44a112	- Uga.SYS	fcc90000	2c4730bc	- Msfs.SYS
fccA0000	2c7d36ee	- Npfs.SYS	fccb0000	2c3609c7	- Ntfs_Rec.SYS
fccd0000	2c87bfe0	- NDIS.SYS	fcc00000	2c87c067	- lance.sys
fcde0000	2c360a2b	- TDI.SYS	fcce0000	2c7ab336	- nbfs.sys
fcdf0000	2c475d75	- streams.sys	fcdf0000	2c545d8d	- ubnh.sys
fcdf0000	2c545df3	- mcsxns.sys	fcdf0000	2c360a32	- netbios.sys
fcdf0000	2c473129	- Parallel.sys	fcdf0000	2c473132	- Serial.SYS
fcdf0000	2c8cdc80	- mup.sys	fcdf0000	2c360a35	- SMBRSUP.SYS
fcdf0000	2c7d36bf	- rdr.sys	fcdf0000	2c8f6901	- browser.sys
fcdf0000	2c4b2868	- afd.sys	fcdf0000	2c7ab1d0	- srv.sys

Address	dword	dump	Build [v1.528]	- Name
ff68cb64	80102090	80102090	00000000	ff68ccac 8014fb71 ff68cb94 - ntoskrnl.exe
ff68cb70	8014fb71	8014fb71	ff68cb94	8010fdea ff68cb9c 00000000 - ntoskrnl.exe
ff68cb78	8010fdea	8010fdea	ff68cb9c	00000000 ff68cb9c ff68cf60 - ntoskrnl.exe
ff68cba0	8016483a	8016483a	ff68ce2c	ff68cf60 ff68cc60 ff68cc40 - ntoskrnl.exe
ff68cbb8	8016484e	8016484e	ff68cf60	ff68cc44 8015a43a ff68ce2c - ntoskrnl.exe
ff68cbc4	8015a43a	8015a43a	ff68ce2c	ff68cf60 ff68cc60 ff68cc40 - ntoskrnl.exe
ff68cbd8	8010fd98	8010fd98	00000004	ffbd700c 80102090 ff68cc14 - ntoskrnl.exe
ff68cbe4	80102090	80102090	ff68cc14	00000000 00000001 80000003 - ntoskrnl.exe
ff68cc00	80102204	80102204	00000003	00000001 ff68ccac 00000000 - ntoskrnl.exe
ff68cc24	80102090	80102090	00000000	0a722600 8013b6d4 ff68ce2c - ntoskrnl.exe
ff68cc30	8013b6d4	8013b6d4	ff68ce2c	ff68b000 ff68cf90 ff68ce4c - ntoskrnl.exe
ff68cc48	8013b7aa	8013b7aa	ff68ce2c	ff68cc60 00000004 ffbd700c - ntoskrnl.exe
ff68cc5c	80102090	80102090	00010017	00000000 80407144 00000008 - ntoskrnl.exe
ff68cc68	80407144	80407144	00000008	00000046 00000246 00000000 - hal.dll
ff68cc7c	80102205	80102205	00000008	00000282 00000003 ffbd700c - ntoskrnl.exe

```
Kernel Debugger Using: COM2 (Port 0x2f8, Baud Rate 19200)
Restart you computer. If this message reappears, do not restart.
Contact you system administrator or technical support group, and/or
peripheral device vendor.
```

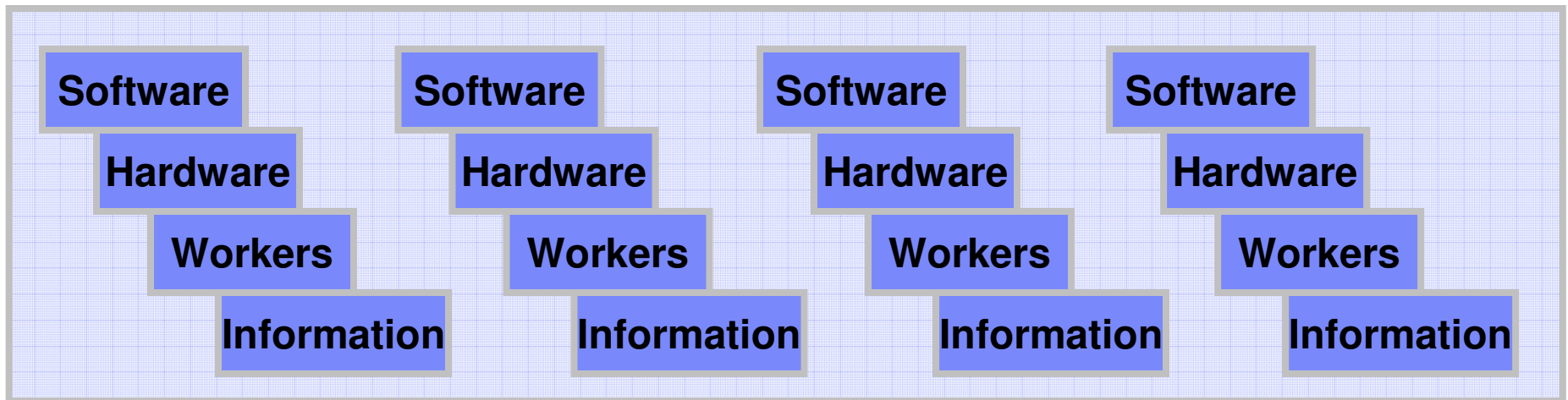
A System

- ... is made up of
 - ▶ Software
 - ▶ Hardware
 - ▶ Workers (people)
 - ▶ Information (data)



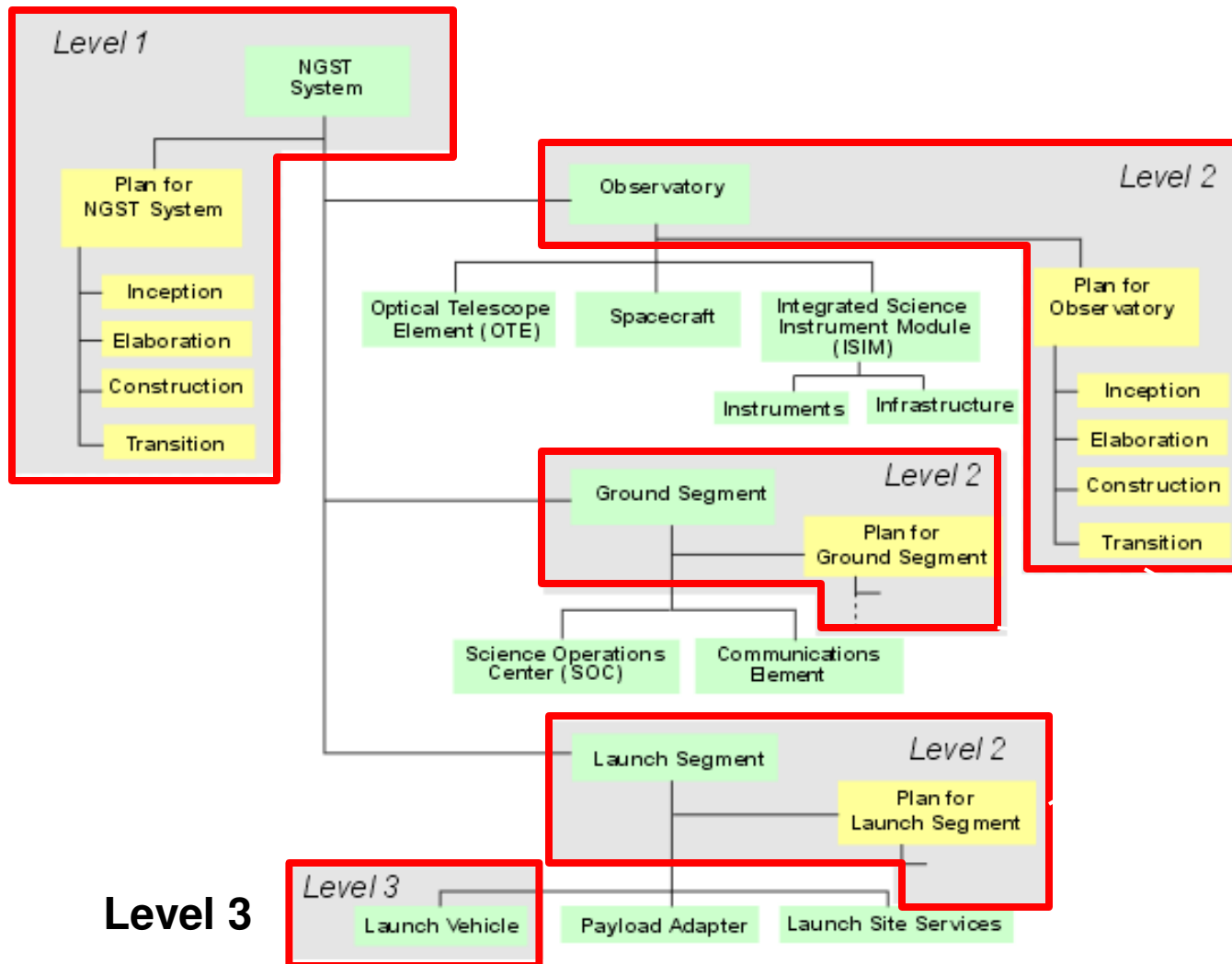
A System of Systems

- 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



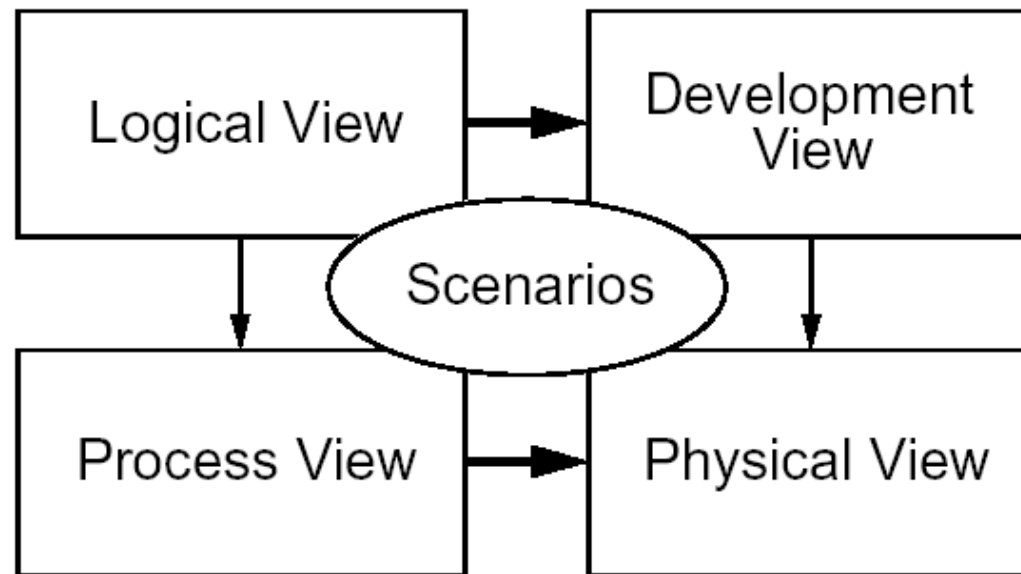
Level 2

Level 3

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)

Viewpoint Level	Worker	Logical	Information	Physical	Process
Context					
Analysis					
Design					
Implementation					

Agenda

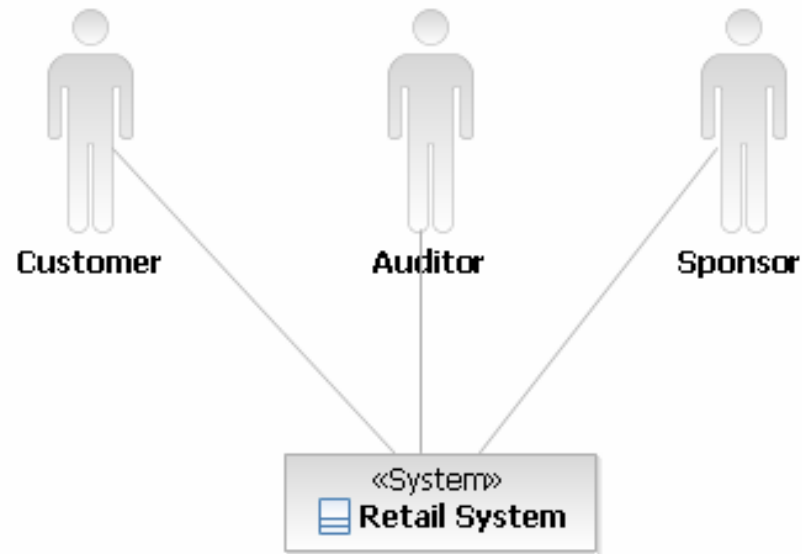
- What is a large-scale system?
- The importance of architecture
- Large-scale architecture in practice
- ➡ A worked example
 - Summary

An example

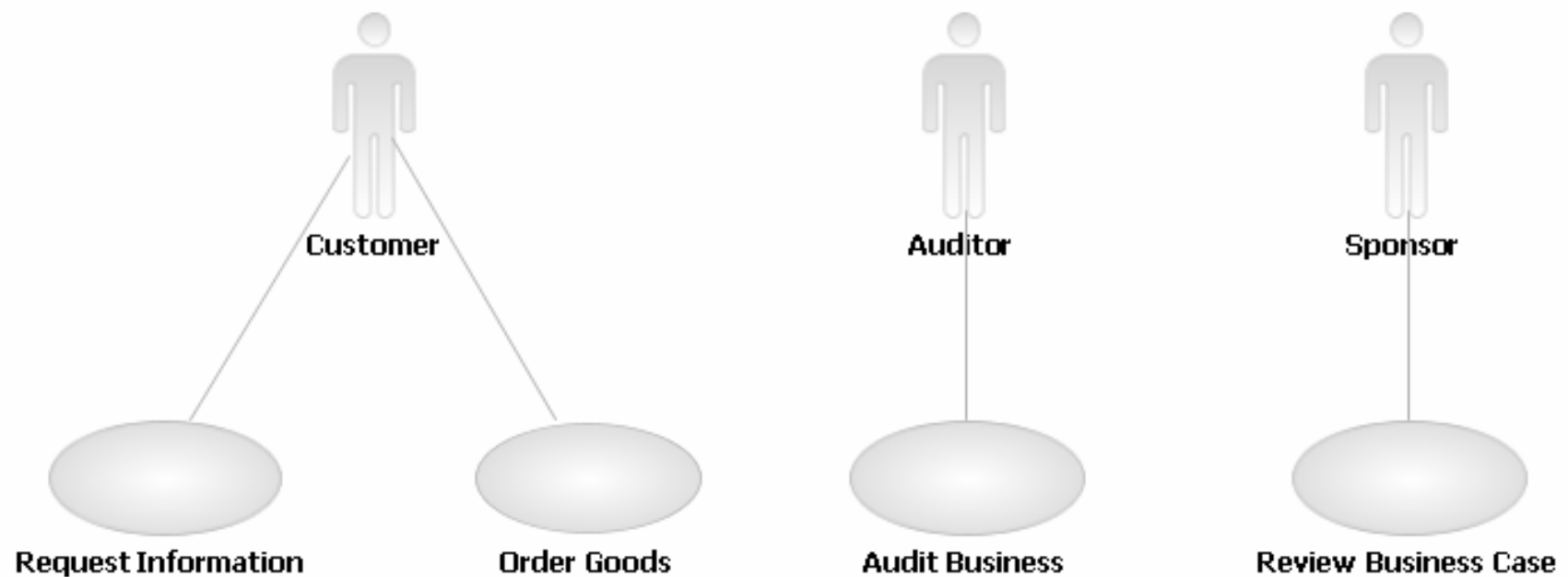
- A retail store
- Selling books, videos, DVDs, music CDs, etc.

Is a sales clerk inside or
outside the system?

Level 1: Context Diagram (initial)



Level 1: Use-Case Model



Level 1: Use-Case Model

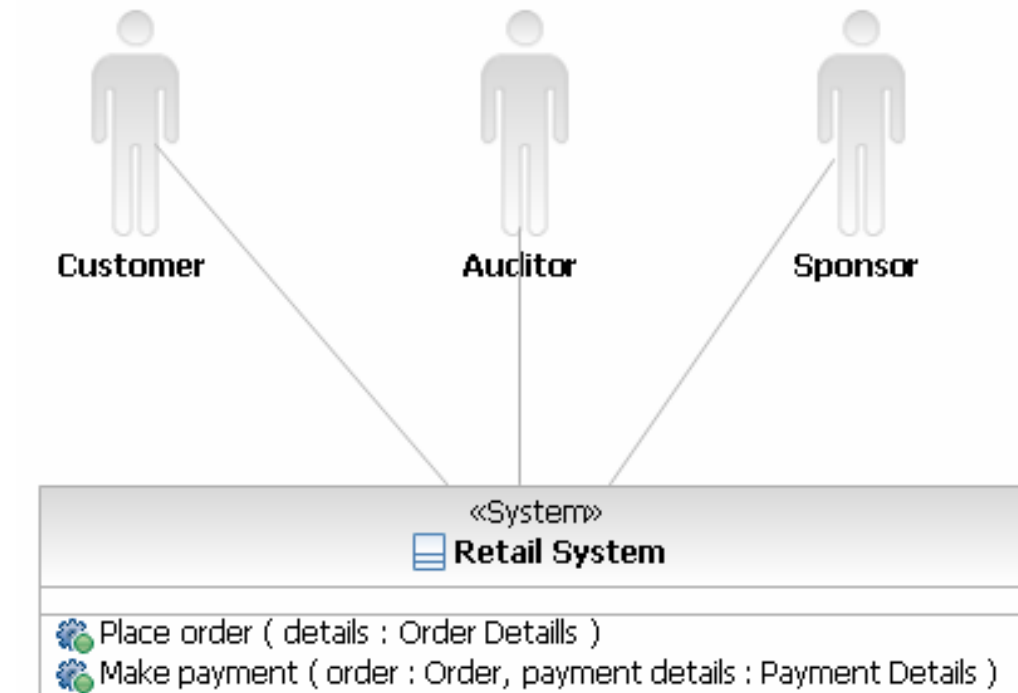
- Basic Flow of the “Order Goods” Business Use Case
 - ▶ The use case starts when the Customer initiates the placing of an Order for Products.
 - ▶ An appropriate Order is placed that contains the Products to be purchased, along with the relevant quantity of each Product. The Customer receives the ordered 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

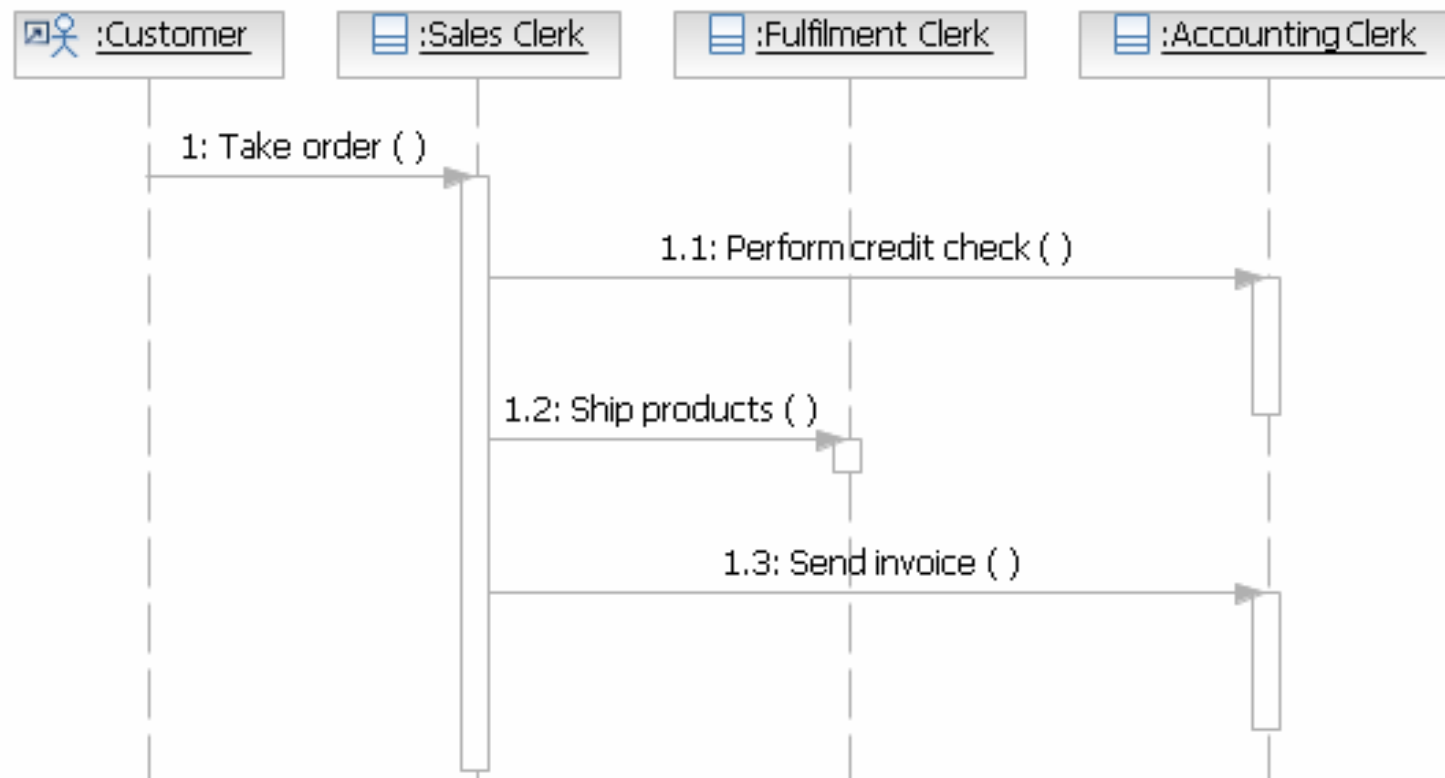


Level 1: Context Diagram (partial)



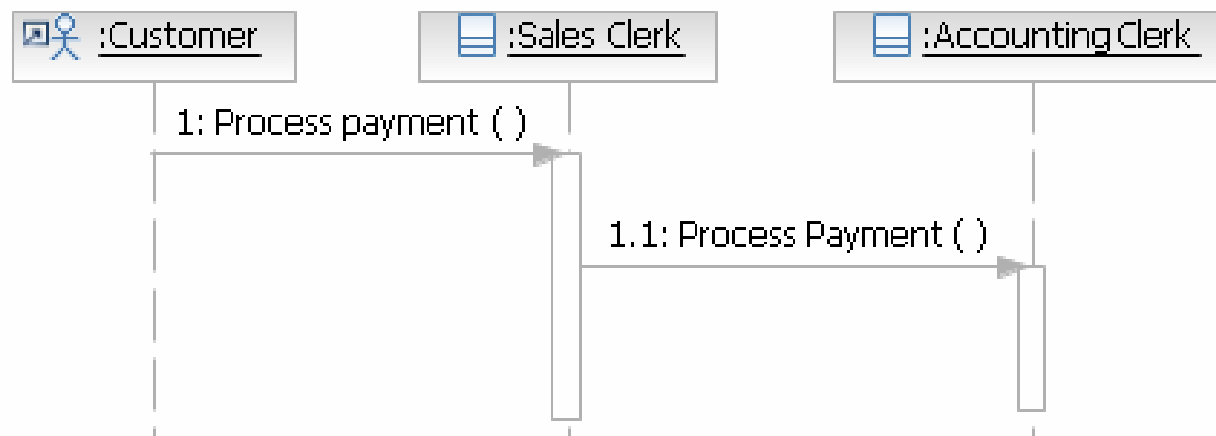
Level 1: Operation Realization

- For “Place order” operation
- The system is treated as a “white box”

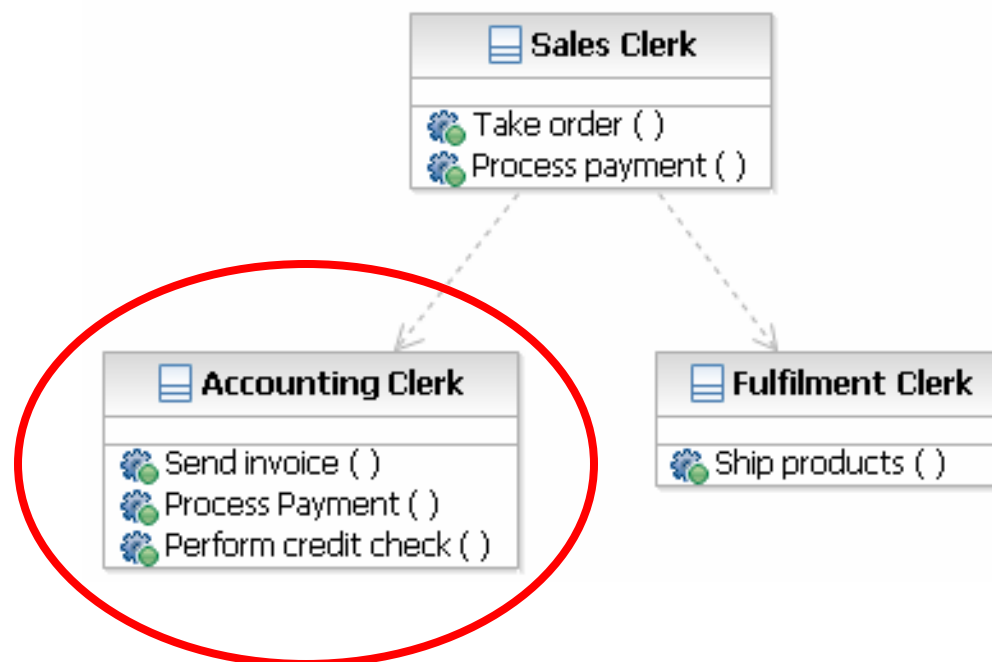


Level 1: Operation Realization

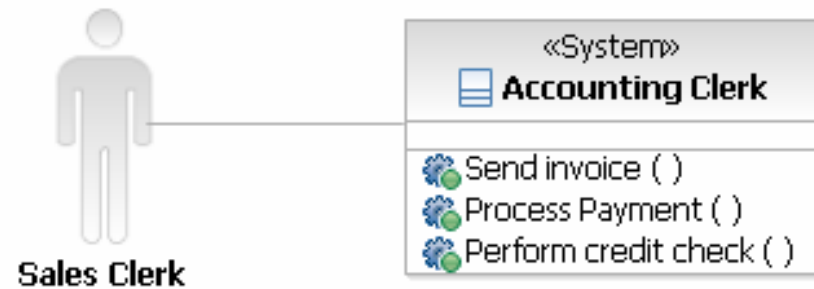
- For “Make Payment” operation



From Level 1 to Level 2

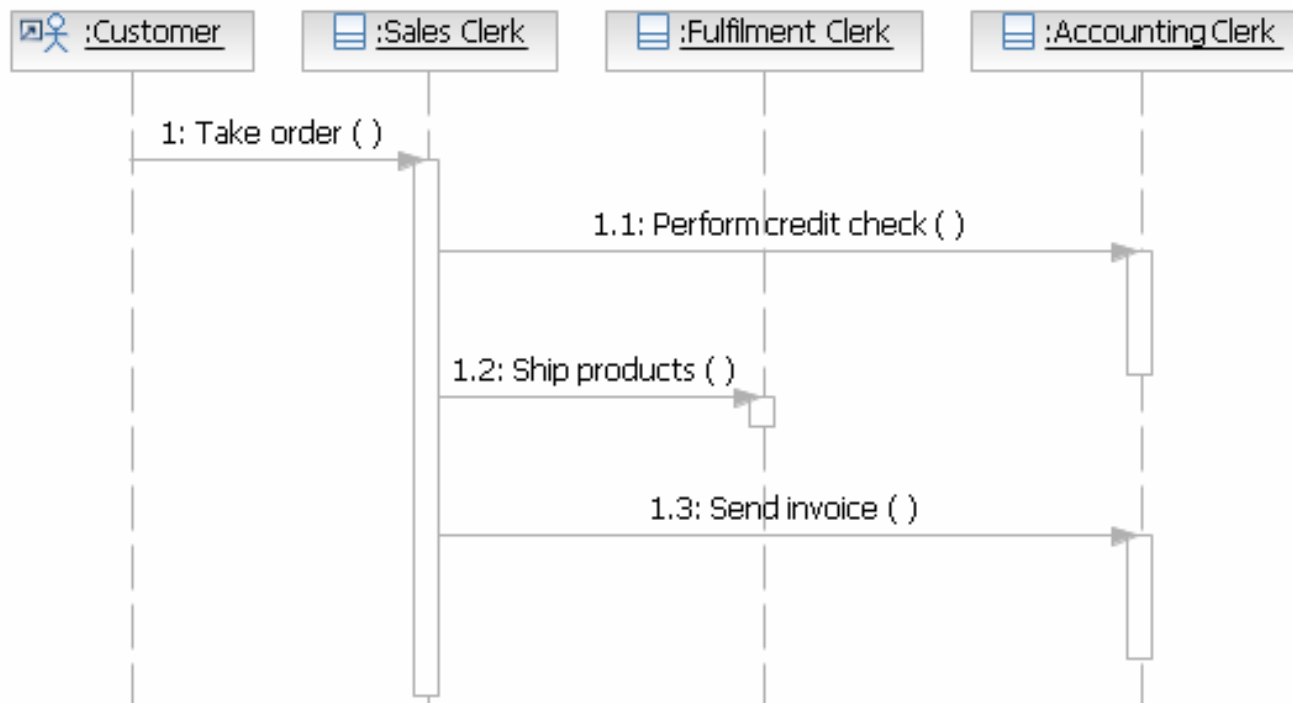


Level 2: Context Diagram



Level 1: Operation Realization

- What about non-functional requirements?
- What about other viewpoints (other than logical or worker)?



Describing an Architecture – Cantor (RUP-SE)

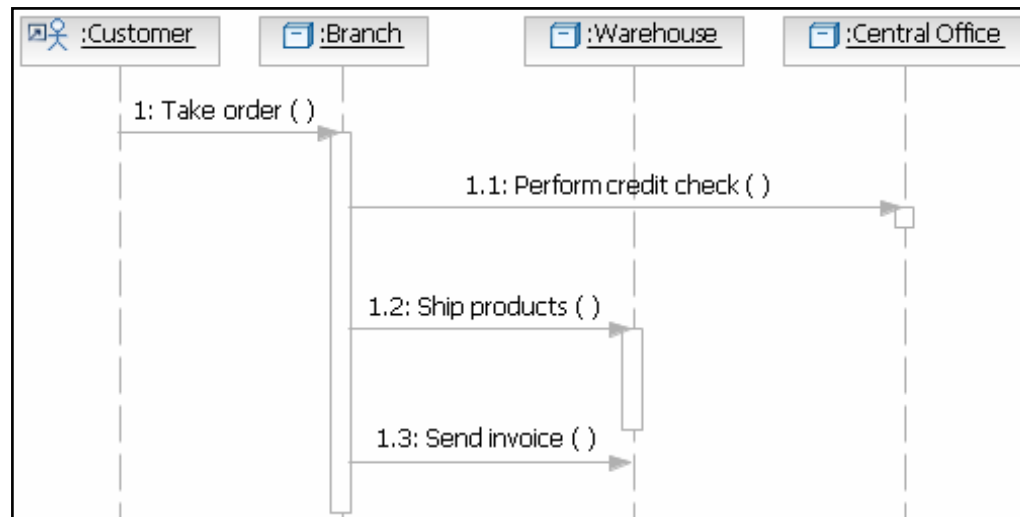
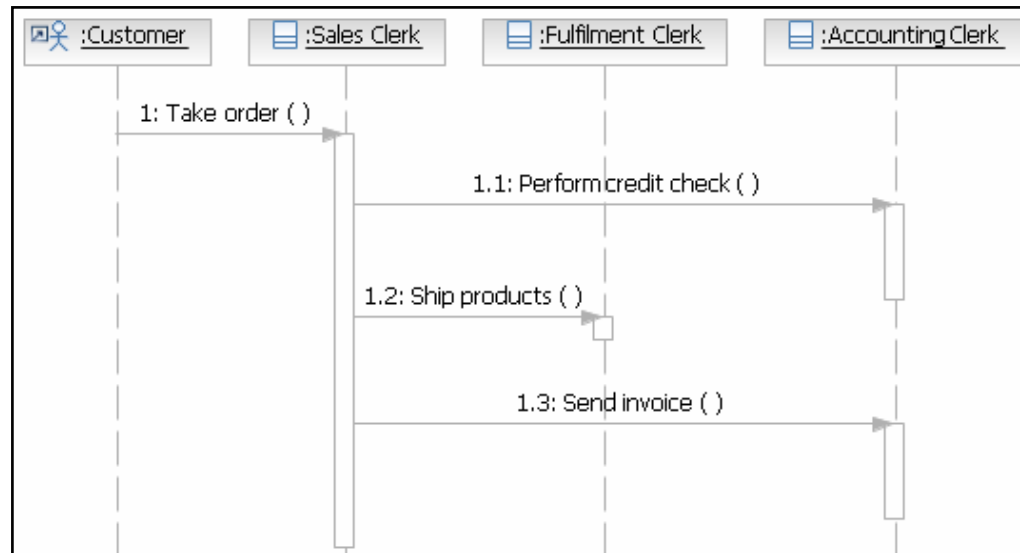
Viewpoint Level	Worker	Logical	Information	Physical	Process
Context					
Analysis	Subsystem	Subsystem		Locality	
Design					
Implementation					

Level 1: Operation Realization

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

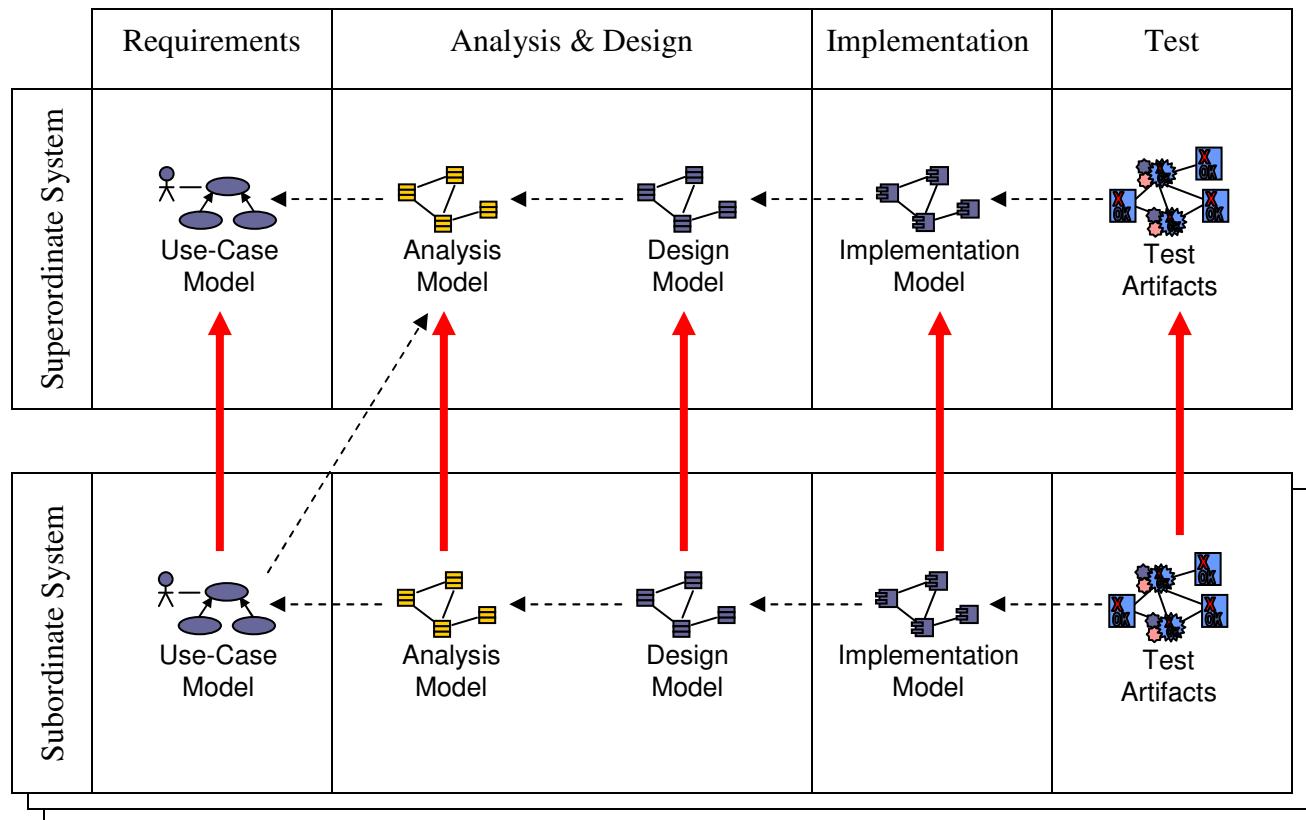
Step	Action Performed	Subsystem	Locality	Budgeted Requirements
1	The order details are taken	Sales Clerk	Branch	60 seconds
2	A credit check is performed	Accounting Clerk	Central Office	10 seconds
3	The products are shipped to the customer	Fulfilment Clerk	Warehouse	1 day
4	An invoice is sent to the customer	Accounting Clerk	Warehouse	1 day

Joint Realization



The “System of Interconnected Systems” Pattern

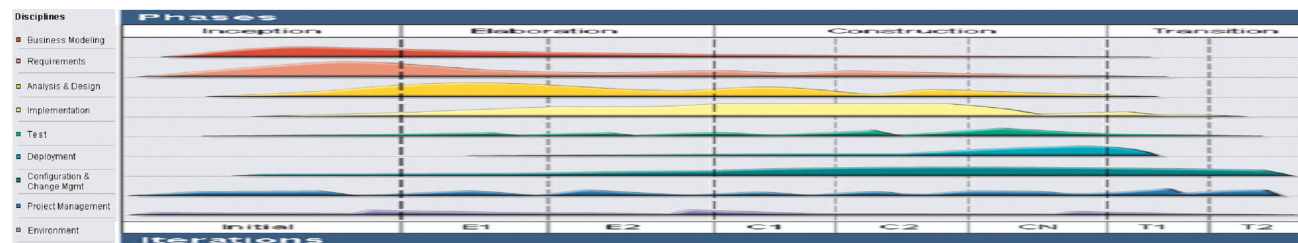
- An example using the Rational Unified Process



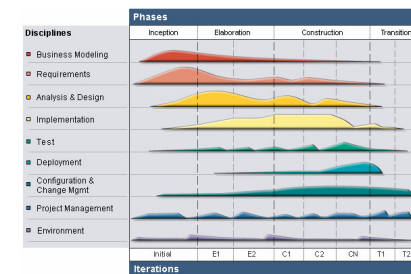
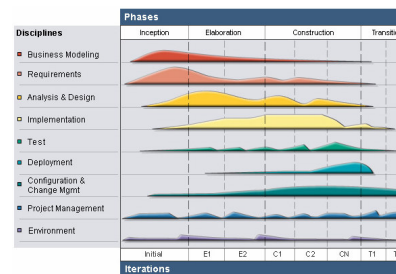
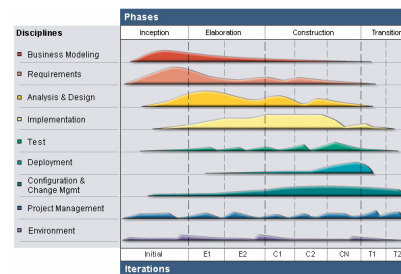
Programme / Project Governance

- Programme concerns
 - ▶ Alignment of projects within a programme

Programme



Projects



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?
 - ▶ <http://www-128.ibm.com/developerworks/rational/library/feb06/eeles/index.html>
- Characteristics of a Software Architect
 - ▶ <http://www-128.ibm.com/developerworks/rational/library/mar06/eeles/index.html>
- The Process of Software Architecting
 - ▶ <http://www-128.ibm.com/developerworks/rational/library/apr06/eeles/index.html>
- The Benefits of Software Architecting
 - ▶ <http://www-128.ibm.com/developerworks/rational/library/may06/eeles/index.html>
- Hardware/software codevelopment using a model-driven systems development (MDSD) approach
 - ▶ <http://www-128.ibm.com/developerworks/rational/library/dec05/cantor/index.html>



Questions



Thank You

Peter Eeles
peter.eeles@uk.ibm.com