Innovate2011

The Premier Software and Product Delivery Event

Software. Everyware.

Why MbSE?

Streamlining the Development of Complex Systems

Hans-Peter Hoffmann, Ph.D.

Chief Systems Methodologist

IBM Rational Software

hoffmape@us.ibm.com

What is Model-based Systems Engineering?

- ***Model-based Systems Engineering*** (MbSE) is a structured a pproach for the development of complex systems across the mechanical, electrical, electronic and software disciplines
  - Helps ensure that all requirements are fulfilled; that functio nal behaviors are realized while non-functional constraints are met
  - Employs **models** as the primary artifacts throughout the sy stems development lifecycle
  - Facilitates improved communication among stakeholders o n a systems development team
  - Provides a disciplined way to manage complexity through a bstraction

Complex systems are everywhere

Aerospace and Defense

e.g.
Flight control
Weapons
Navigation
Guidance
Communications
Autonomous systems
...

Automotive

e.g.
Power train
Safety
Entertainment
Comfort and convenience
Instrumentation
Communications
...

Electronics …

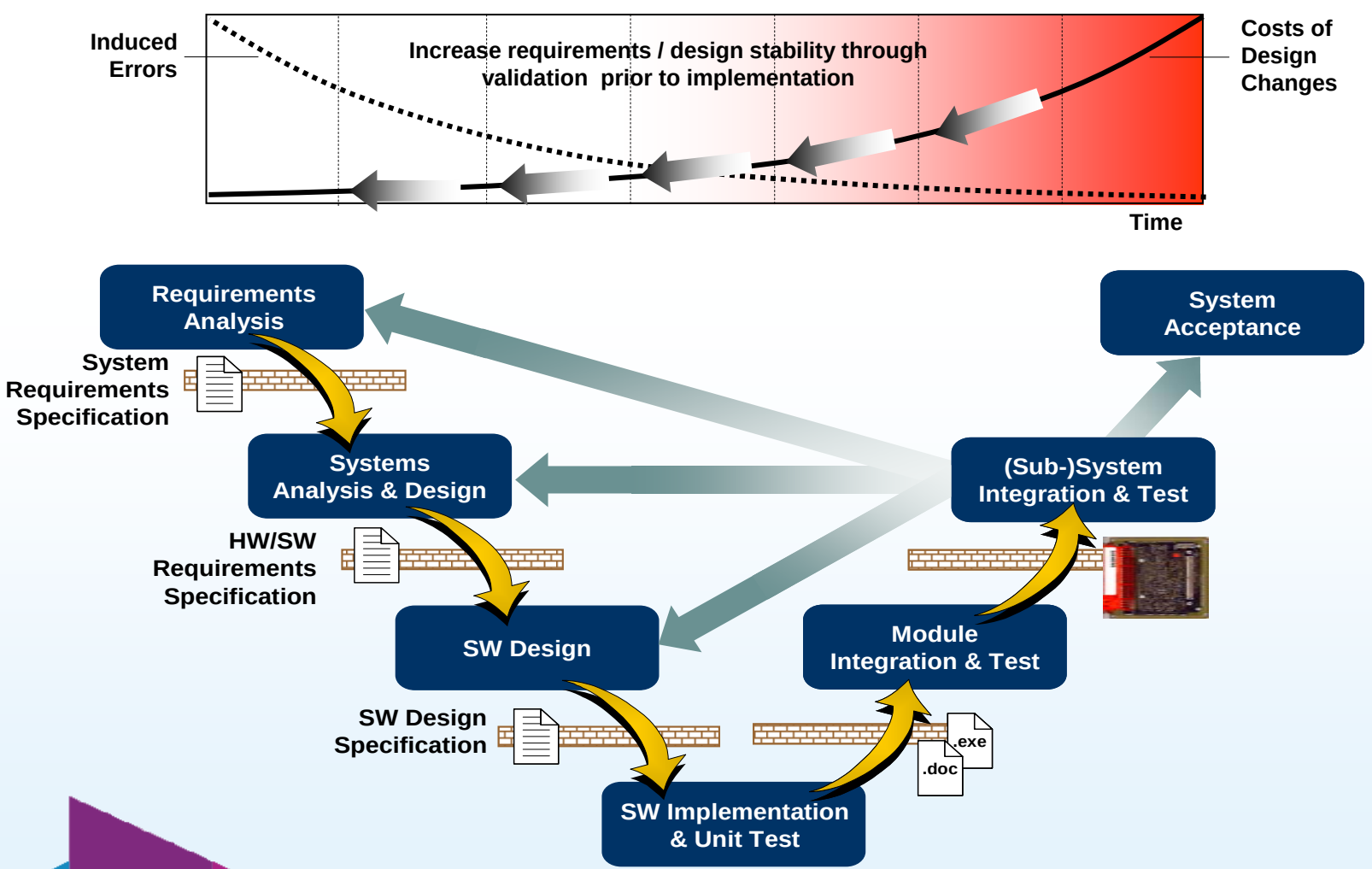
e.g.
Medical
Industrial
Consumer
Transport
Telecommunications
...

Agenda

- **Model-based systems engineering in a model-driven development lifecycle**
- Essential SysML artifacts of the Rational MbSE approach
- Task flow in Rational *Harmony™ for Systems Engineering*
- Deploying MbSE with Rational® *Rhapsody®*
- Documentation of Rational *Harmony™ for Systems Engineering*

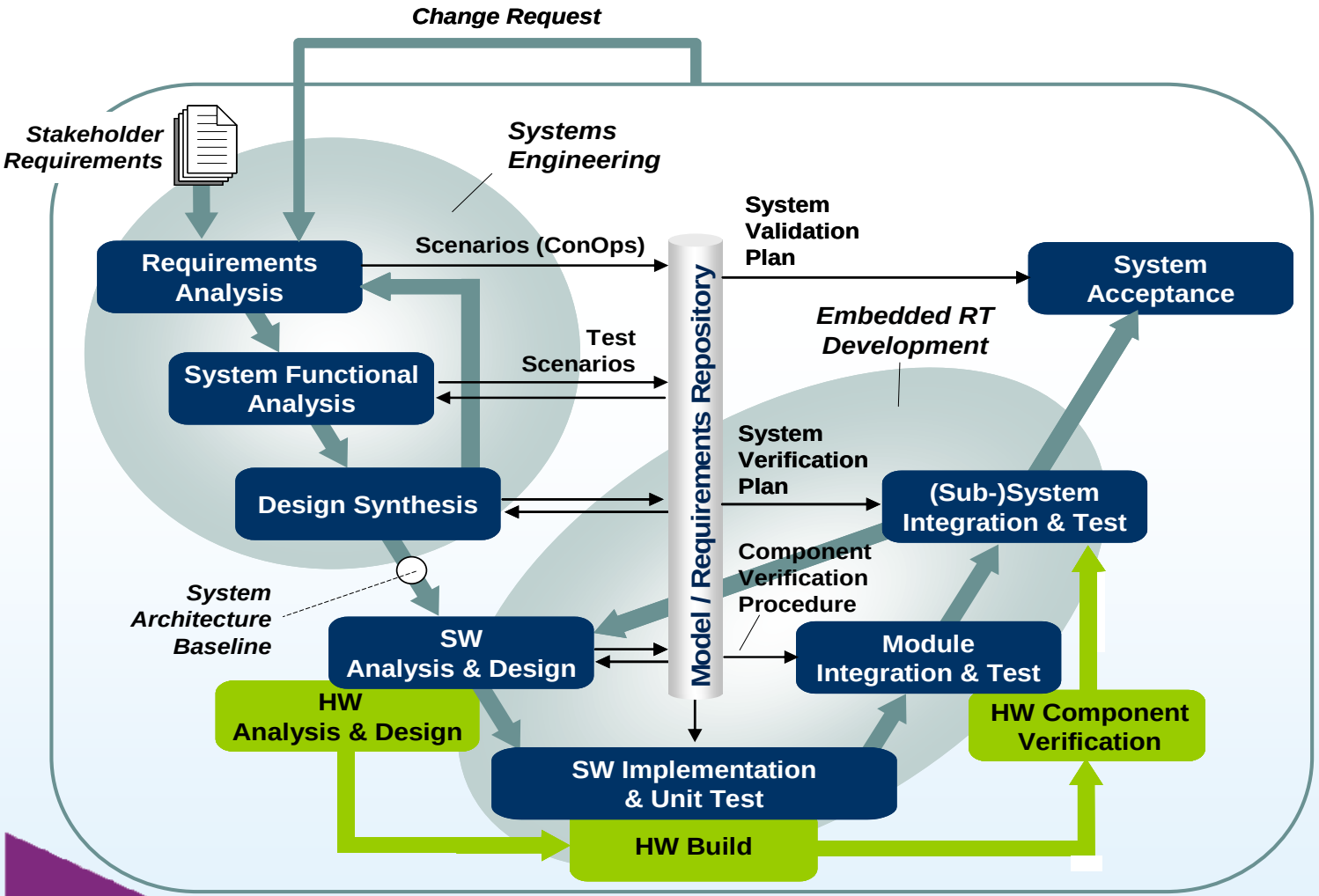
Document driven development of embedded systems

The “Throw-it-over-the-Fence” approach



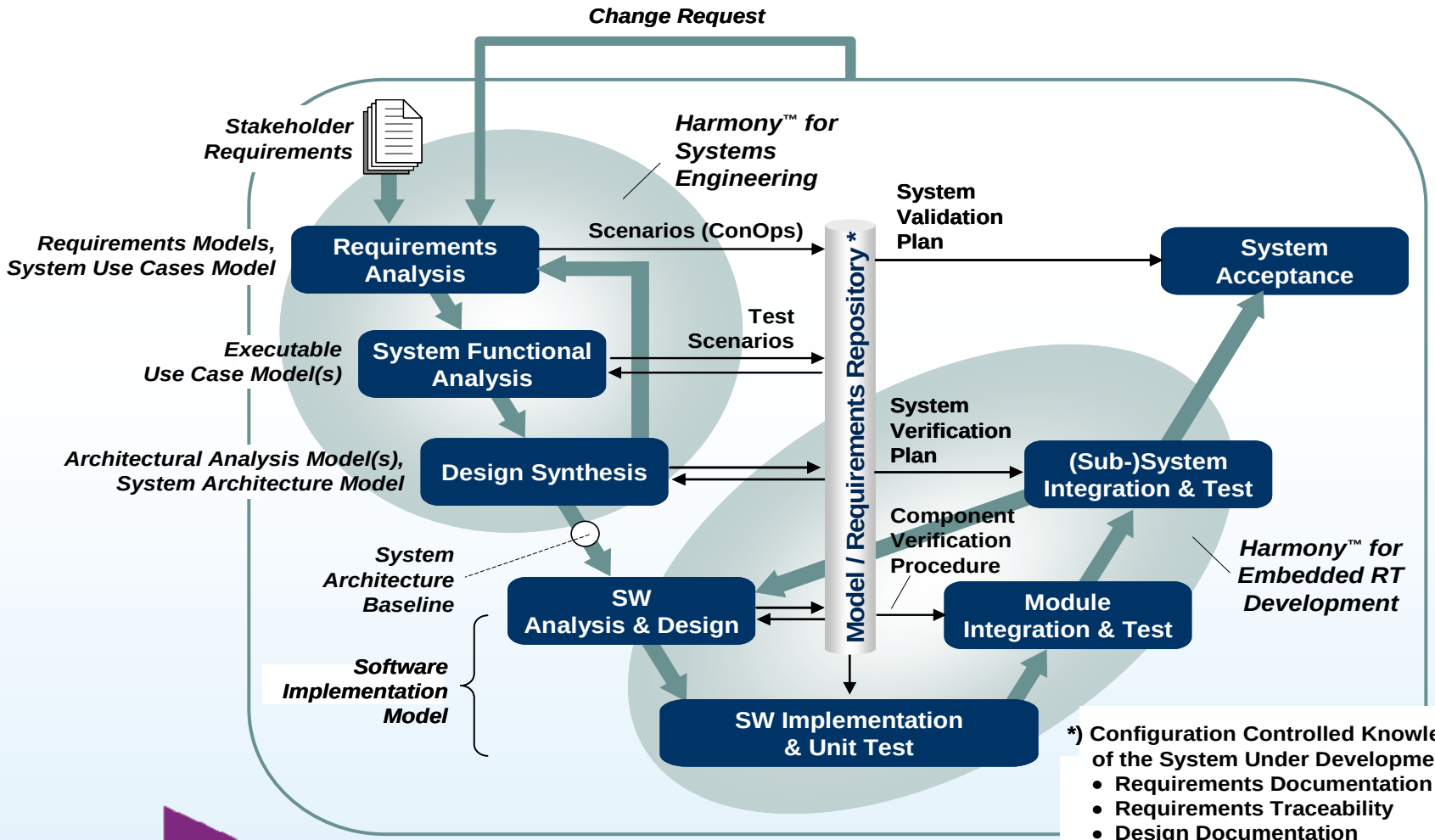
Integrated system / embedded software development

Design iterations in the "V" development lifecycle



Integrated system / software development process

Model-driven development of embedded systems

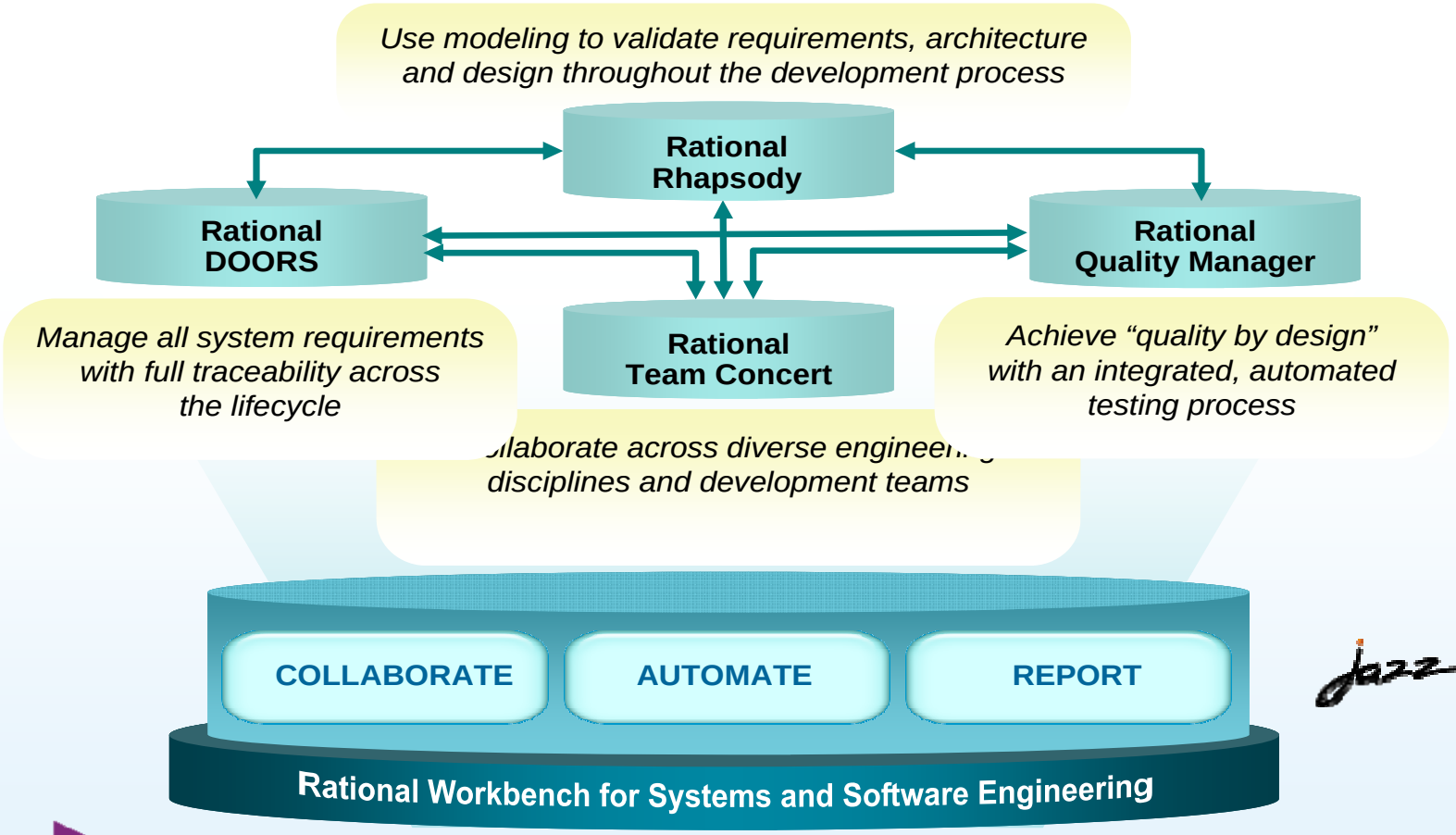


*) Configuration Controlled Knowledge of the System Under Development:

- Requirements Documentation
- Requirements Traceability
- Design Documentation
- Test Definitions

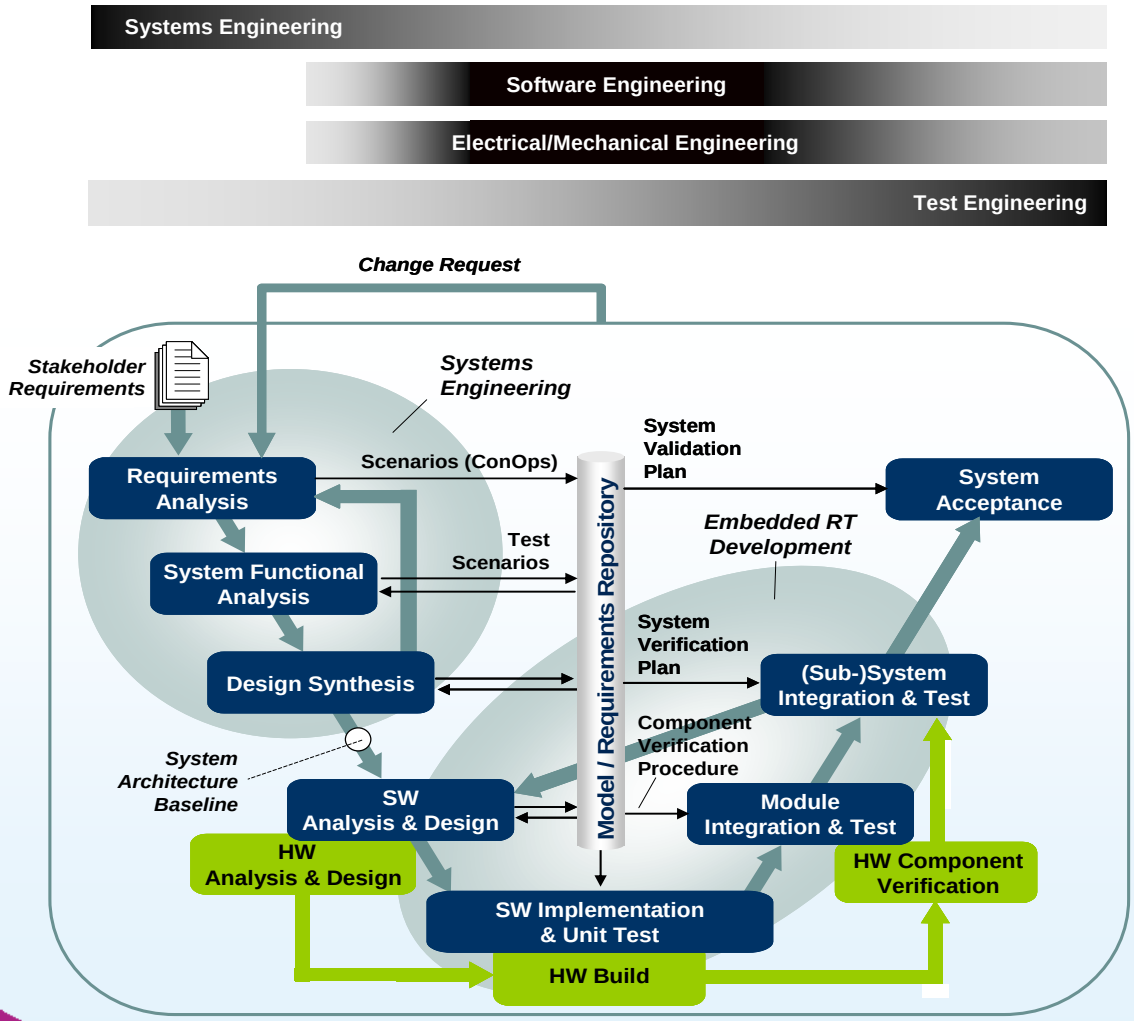
Rational Workbench for Systems and Software Engineering

Built on a core solution set



Integrated system / software development process

Domains involved in the different phases of the model-driven development



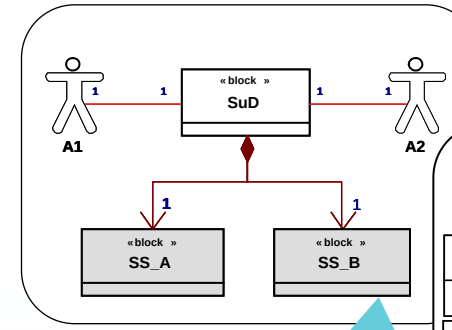
Agenda

- Model-based systems engineering in a model-driven development lifecycle
- **Essential SysML artifacts of the Rational MbSE approach**
- Task flow in Rational *Harmony™ for Systems Engineering*
- Deploying MbSE with Rational® *Rhapsody®*
- Documentation of Rational *Harmony™ for Systems Engineering*

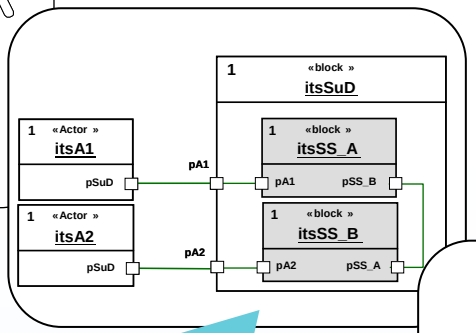
SysML artifacts in Rational *Harmony™ for Systems Engineering*

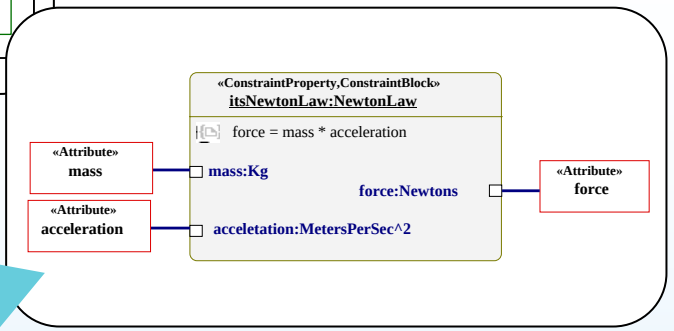
Capturing the static view

Defines structural elements (Blocks) and their relationship



Defines the realization of system structure



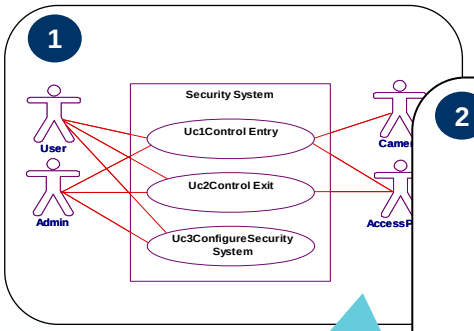


Defines the parametric relationship between system properties

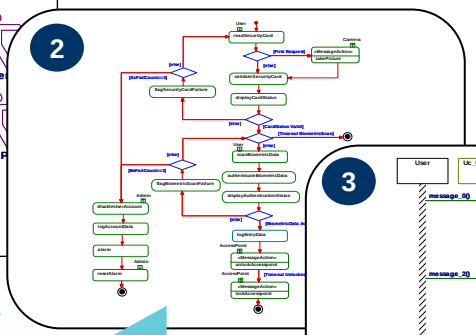
Capturing system behavior in a model-based approach

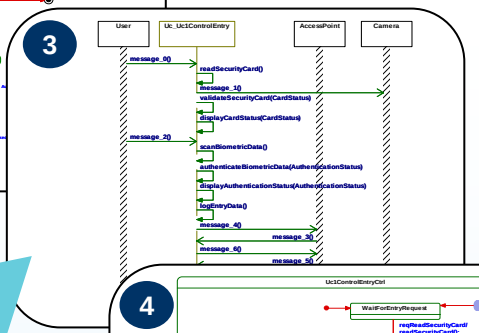
Example: Creating an executable use case model

- Defines system scope
- Groups requirements into Use Cases ("Table of Contents")

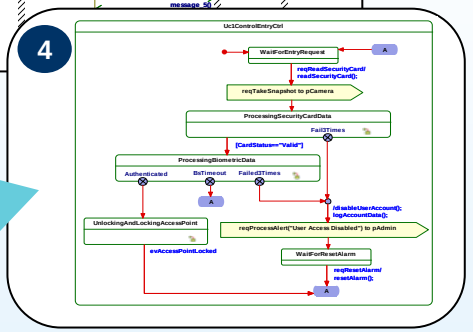


Defines functional flow ("Storyboard") of the Use Case

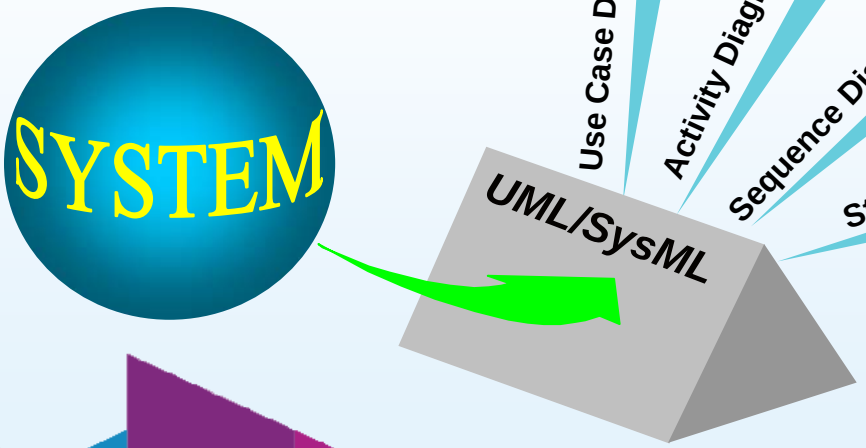




Defines interactions with the environment



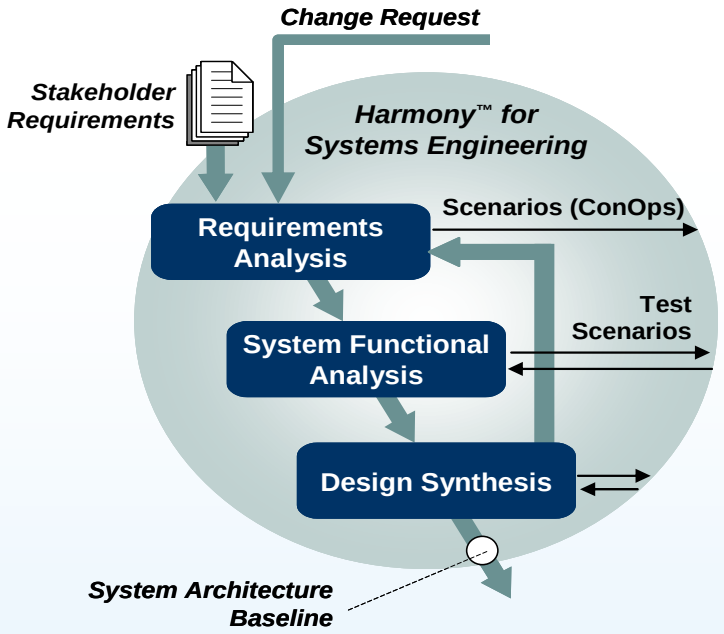
Aggregates AD- and SD-info and adds state-based behavior



Agenda

- Model-based systems engineering in a model-driven development lifecycle
- Essential SysML artifacts of the Rational MbSE approach
- **Task flow in Rational *Harmony™ for Systems Engineering***
- Deploying MbSE with Rational® *Rhapsody®*
- Documentation of Rational *Harmony™ for Systems Engineering*

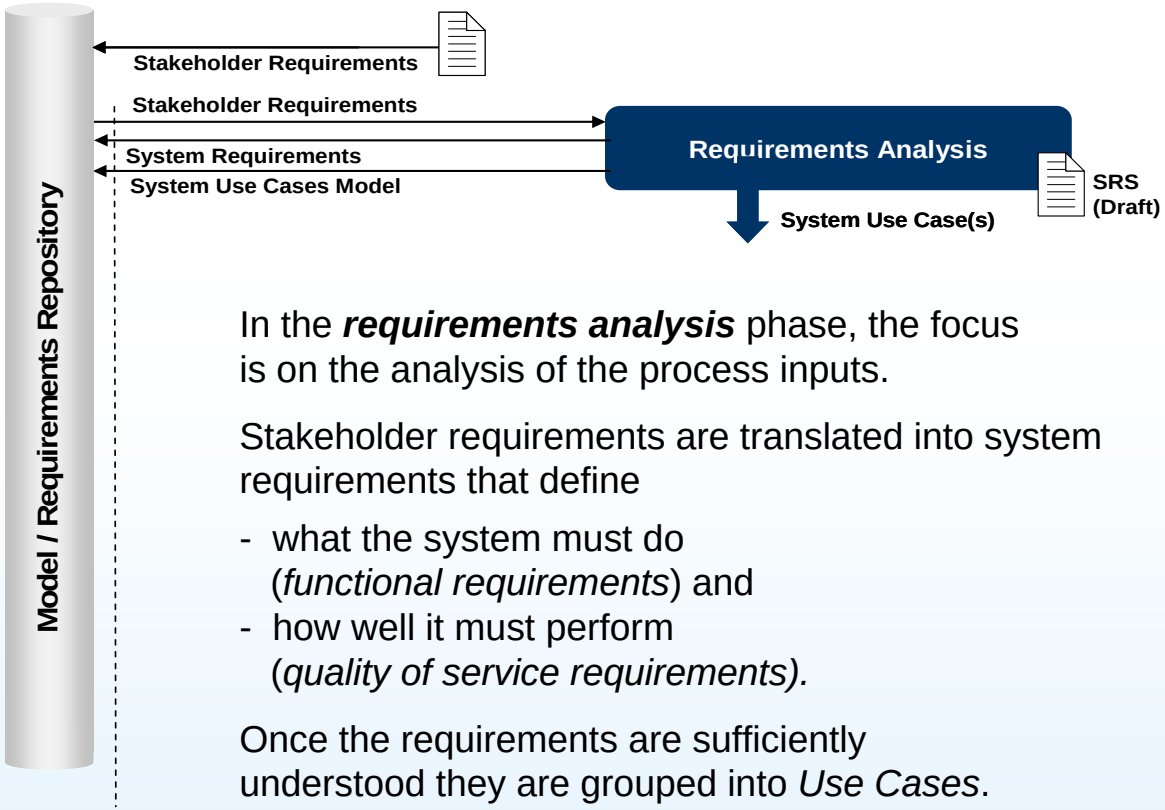
Key objectives of the Rational *Harmony™ for Systems Engineering* workflow



- Identify / derive required system functionality
- Identify associated system states and operational modes
- Allocate required system functionality to a system architecture taking into account non-functional aspects of the requirements

Rational *Harmony™ for Systems Engineering*

Requirements analysis



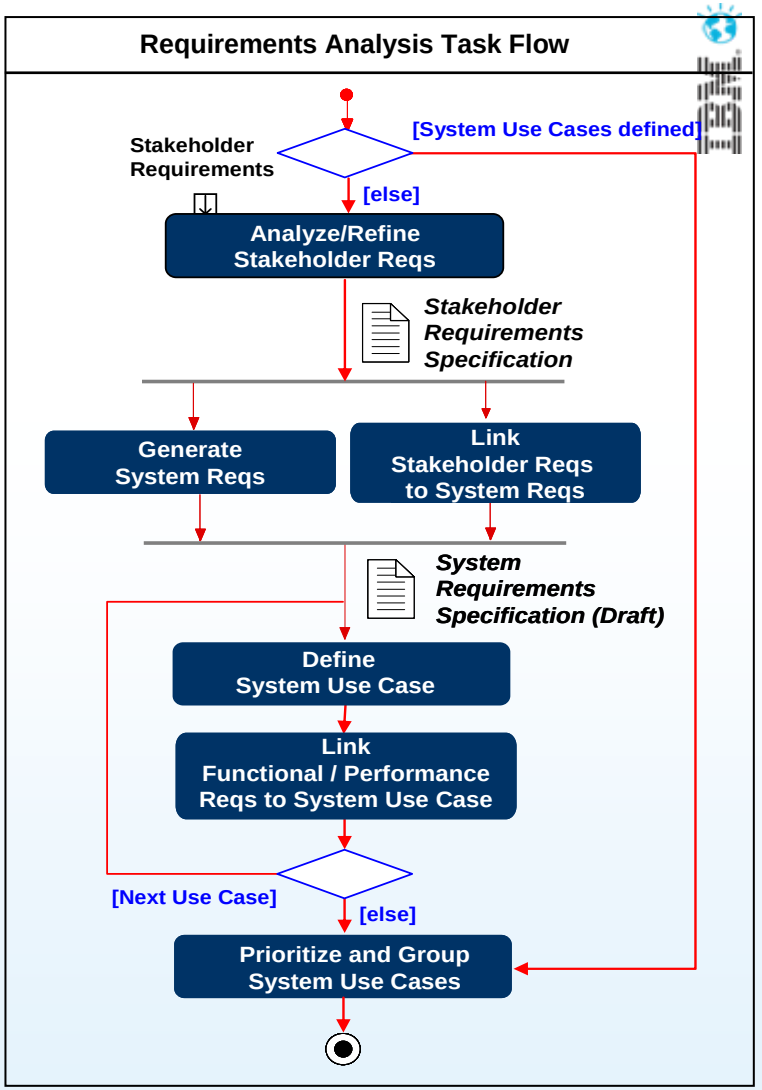
In the ***requirements analysis*** phase, the focus is on the analysis of the process inputs.

Stakeholder requirements are translated into system requirements that define

- what the system must do (*functional requirements*) and
- how well it must perform (*quality of service requirements).*

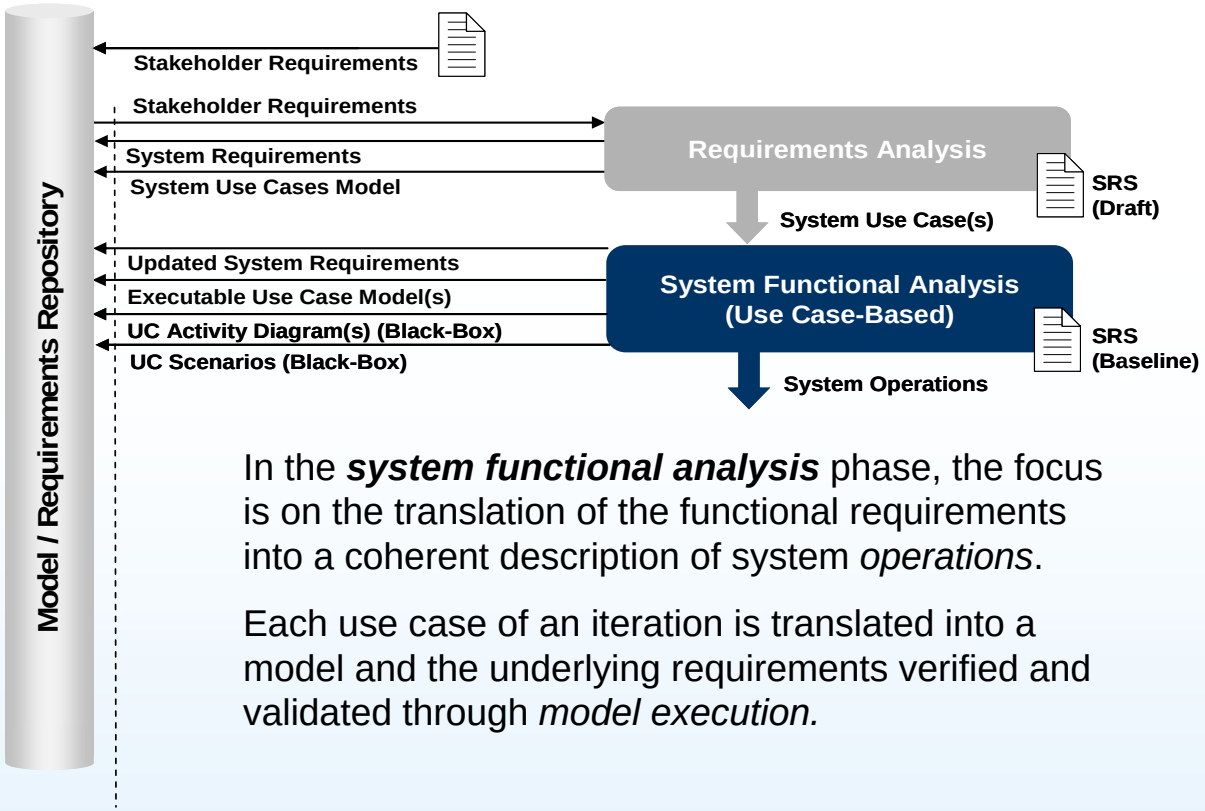
Once the requirements are sufficiently understood they are grouped into *Use Cases*.

Links providing traceability to original requirements



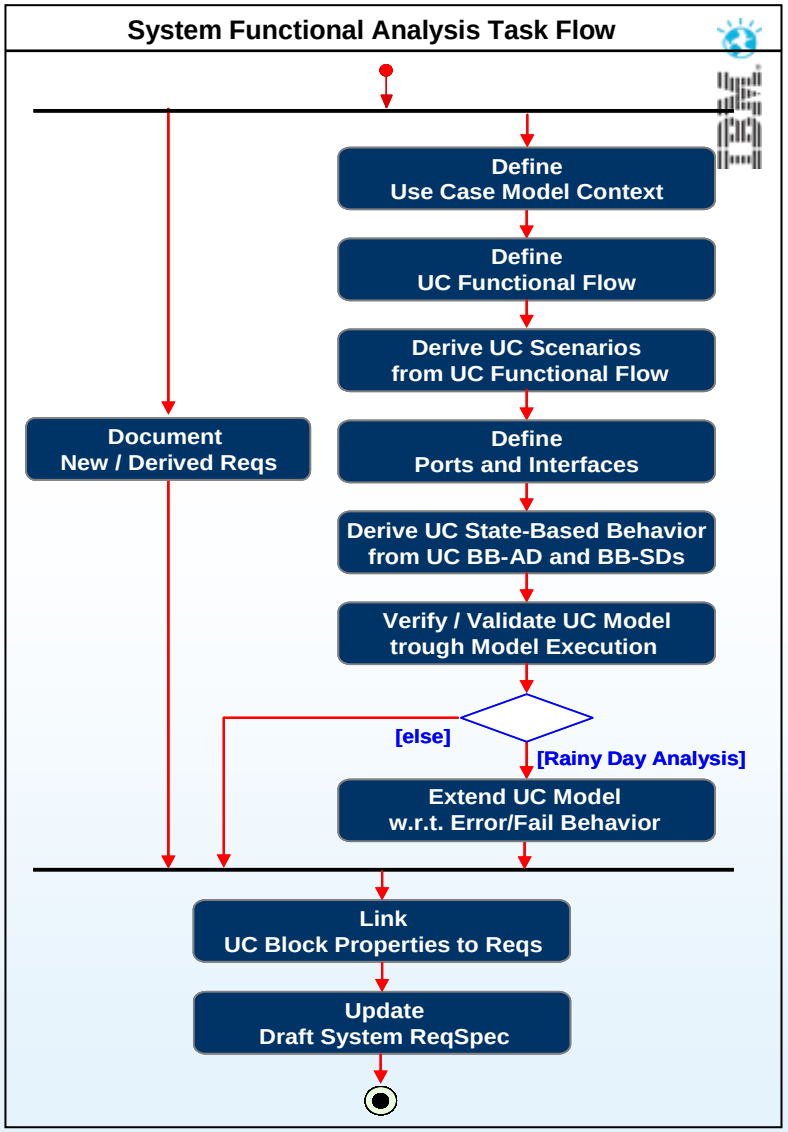
Rational *Harmony™ for Systems Engineering*

System functional analysis

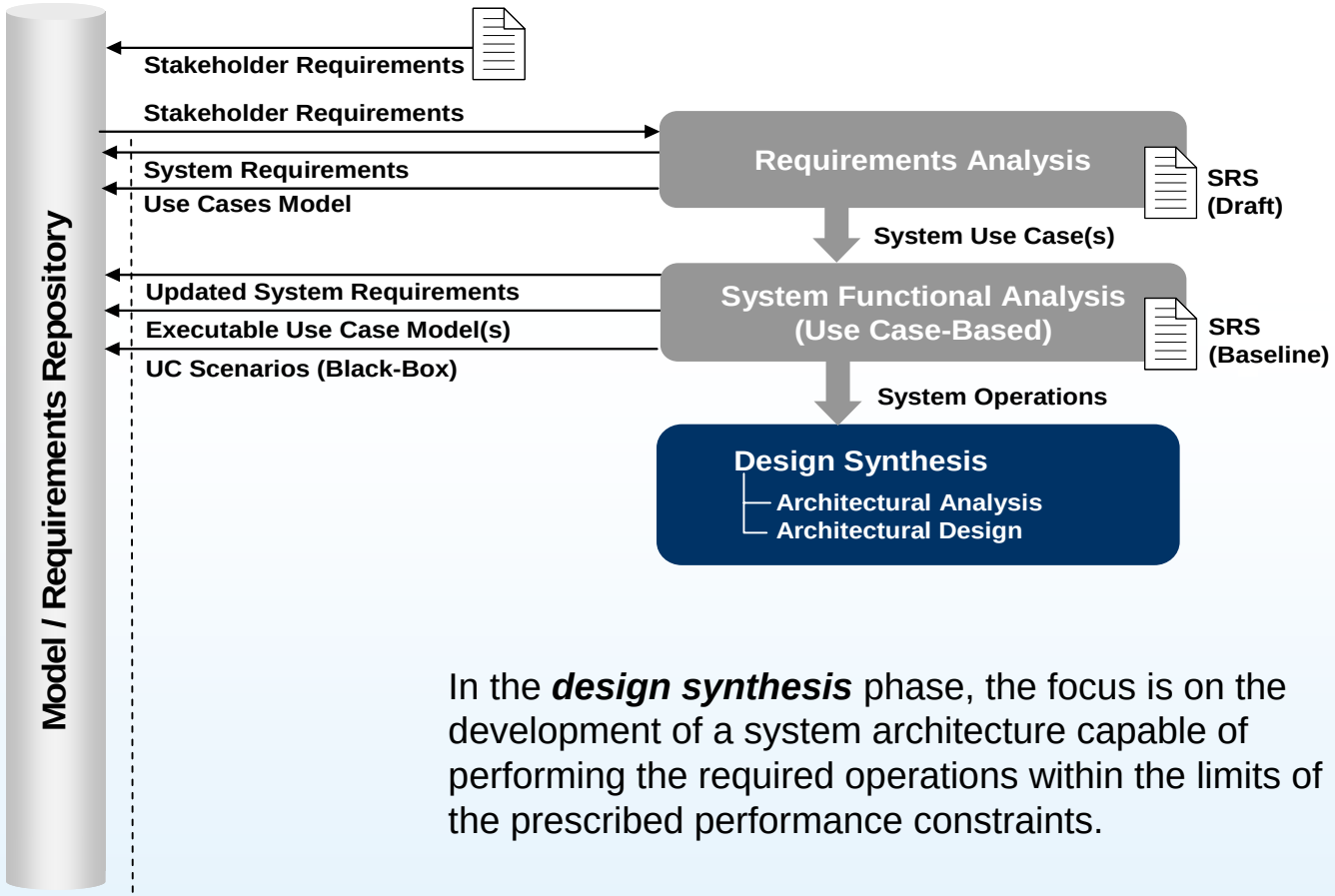


In the ***system functional analysis*** phase, the focus is on the translation of the functional requirements into a coherent description of system *operations*.

Each use case of an iteration is translated into a model and the underlying requirements verified and validated through *model execution.*

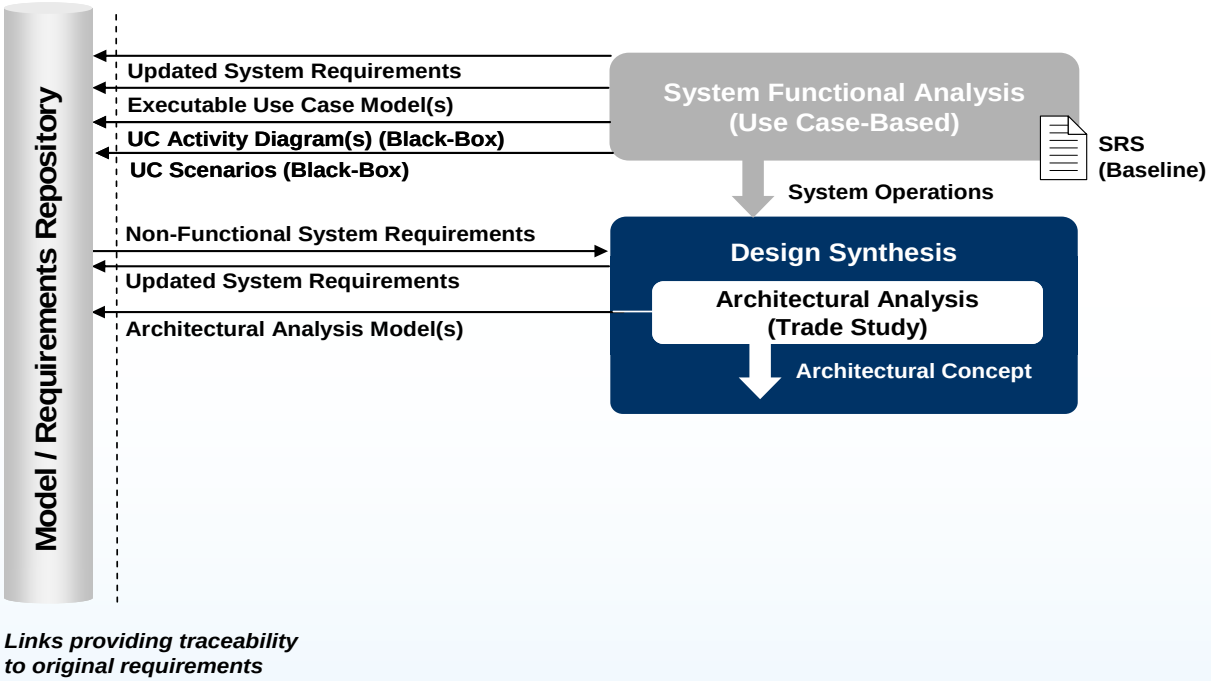


Rational *Harmony™ for Systems Engineering*
Design synthesis



In the ***design synthesis*** phase, the focus is on the development of a system architecture capable of performing the required operations within the limits of the prescribed performance constraints.

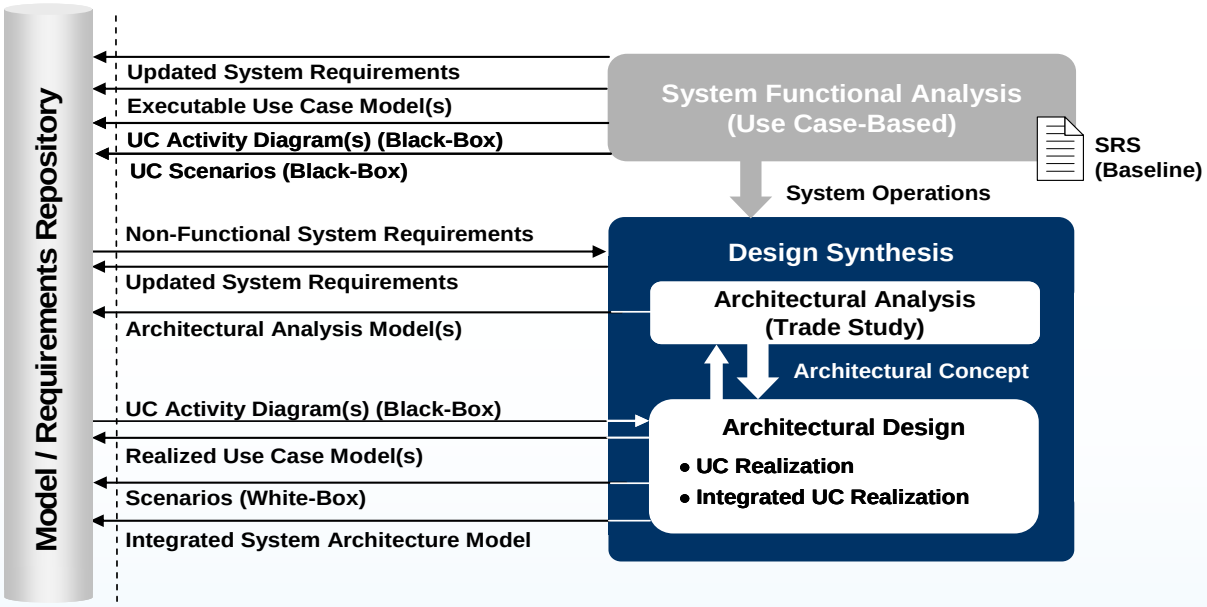
Design synthesis
Architectural analysis



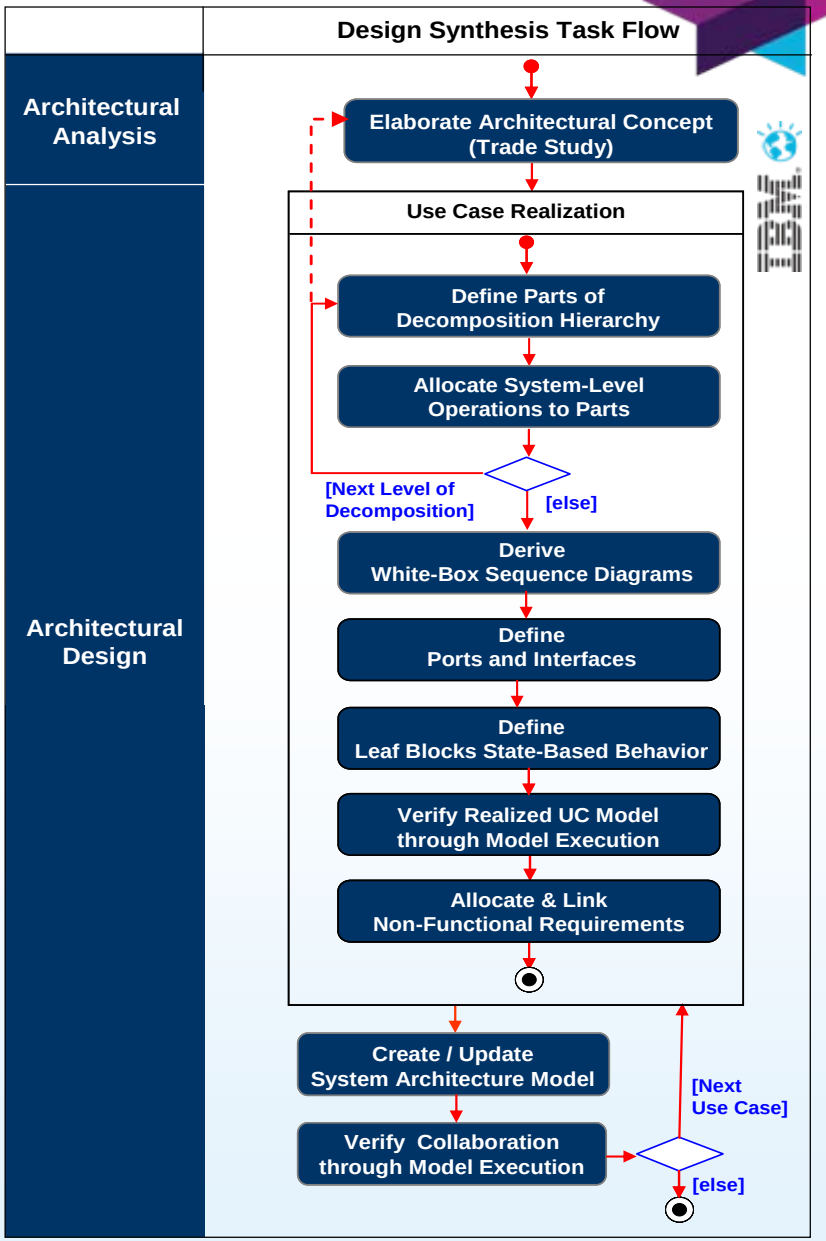
Links providing traceability to original requirements

The objective the ***architectural analysis*** phase is to elaborate the optimum design concept based upon a set of criteria (e.g. *Measures of Effectiveness*, MoEs) that are weighted according to their relative importance.

Design synthesis
Architectural analysis

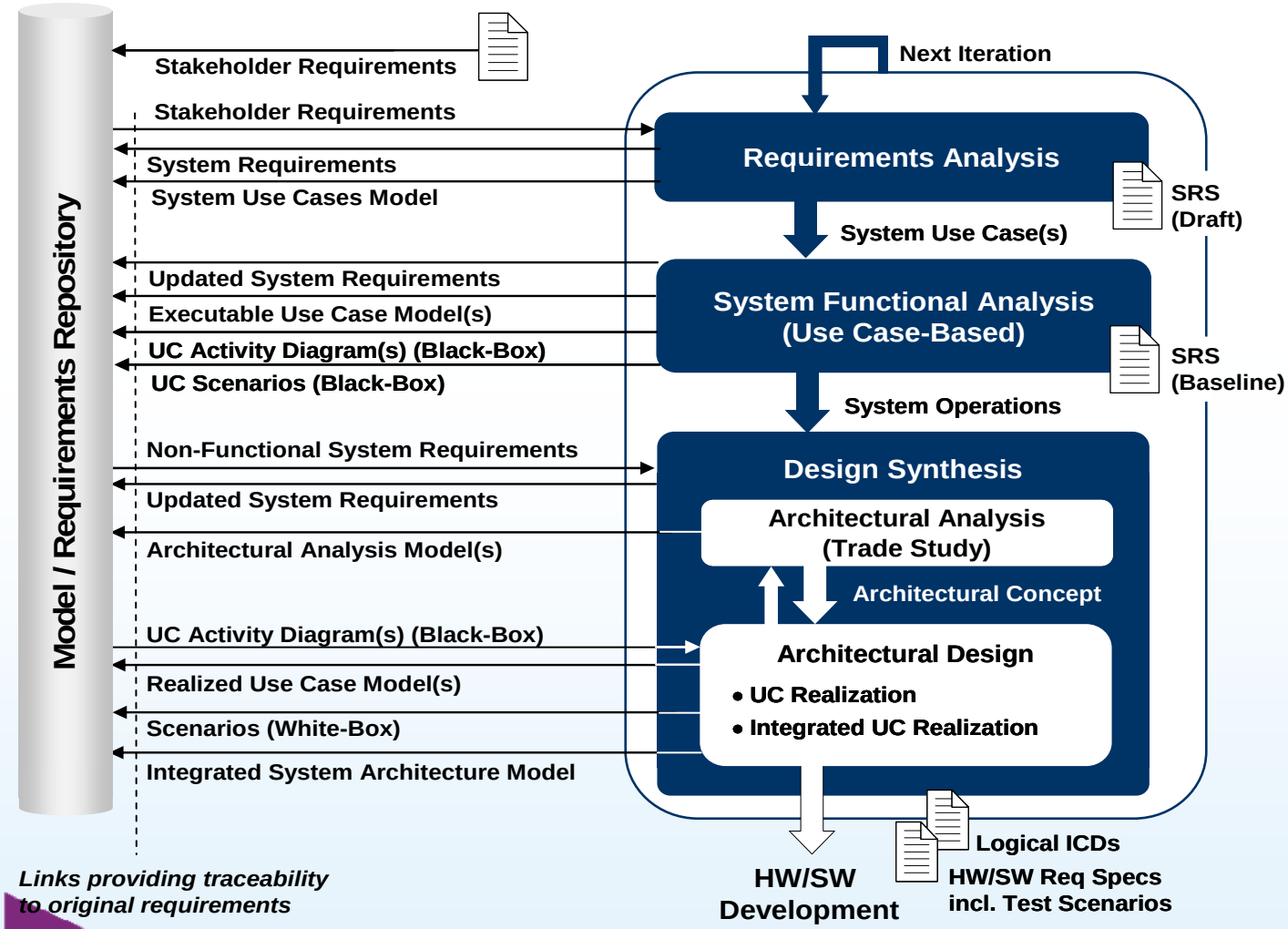


Links providing traceability to original requirements



The focus of the ***architectural design*** phase is on the allocation of the functional requirements (= system operations) and non-functional requirements to an architectural structure.
The architectural design is performed *incrementally* for each use case of an iteration by transitioning from the black-box view to the white-box view – also referred to as ***use case realization***.

Rational *Harmony™ for Systems Engineering*



Systems engineering handoff to the subsequent system development

In a model-driven development the key artifact of the handoff from systems engineering to the subsequent system development is the *baselined* executable model.

This model is the repository from which specification documents (e.g. HW/SW requirements specifications, ICDs, …) are generated.

Scope and content of the hand-off is dependent on the characteristics of the project and the organizational structure systems engineering is embedded.

Systems engineering handoff to the subsequent system development

The hand-off packages typically are composed of baselined executable CI model(s) which contain

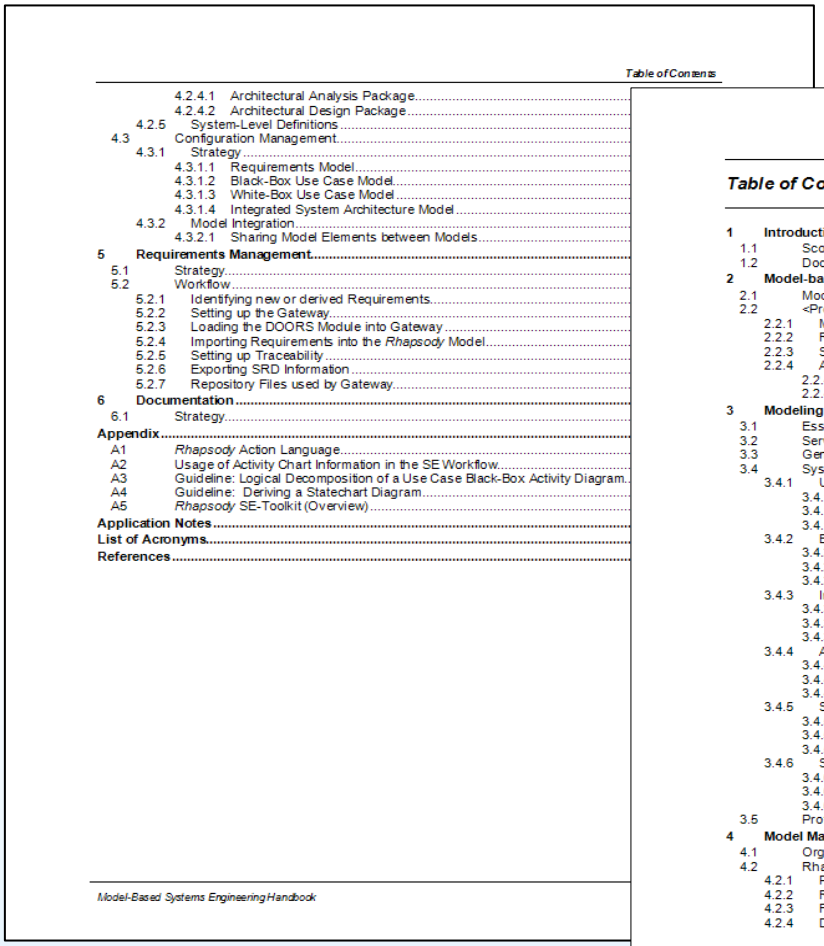
- The definition of allocated operations including their links to the associated system functional and performance requirements
- The definition of the associated state-based behavior, captured in a statechart diagram
- The allocated and linked non-functional requirements
- The definition of ports and *logical* interfaces
- Test scenarios, derived from system-level use case scenarios

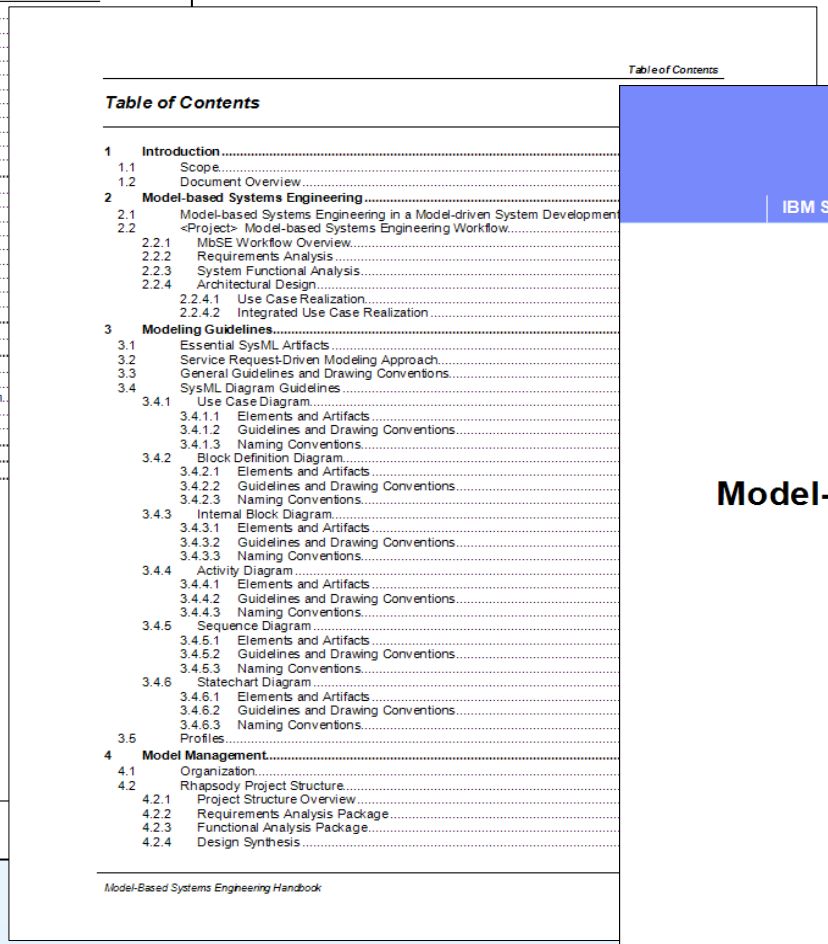
Agenda

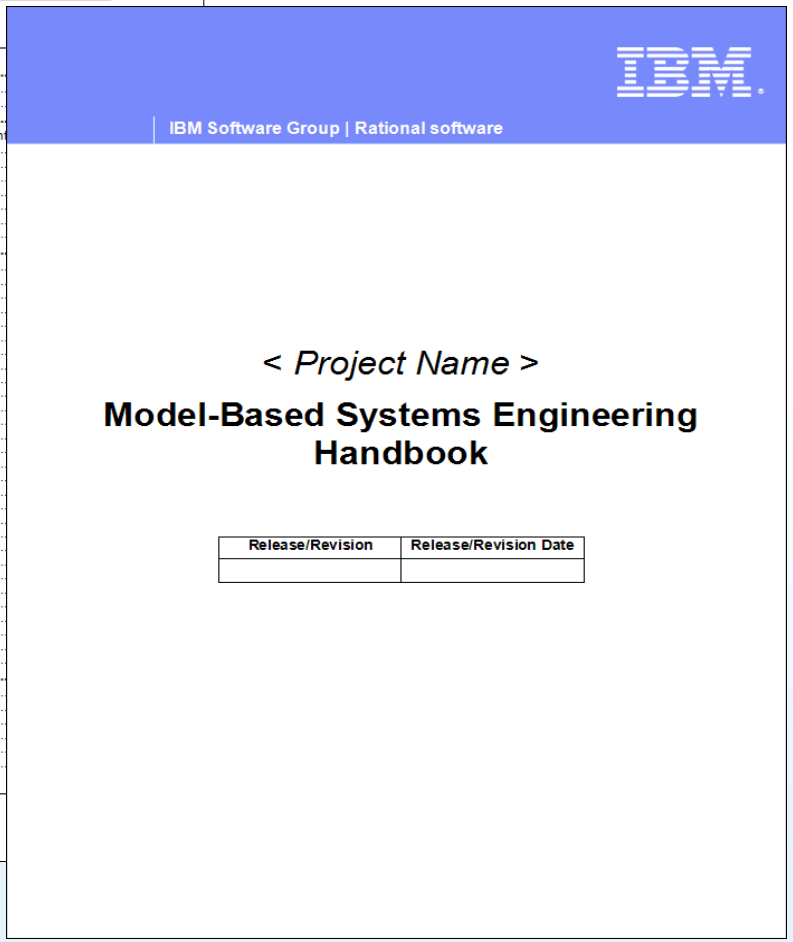
- Model-based systems engineering in a model-driven development lifecycle
- Essential SysML artifacts of the Rational MbSE approach
- Task flow in Rational *Harmony™ for Systems Engineering*
- **Deploying MbSE with Rational® *Rhapsody®***
- Documentation of Rational *Harmony™ for Systems Engineering*

Deploying model-based systems engineering

Project specific model-based Systems Engineering Handbook







2.2 Model-based Systems Engineering Workflow

The model-based systems engineering workflow applied in the *xxxx* project is a specialization of the Harmony/SE workflow outlined in the previous chapter. This chapter outlines the details of the MbSE workflow. It starts with a general overview. Then, for each phase the associated tasks and associated work products are described in detail.

2.2.1 MbSE Workflow Overview

Fig. 2.2-1 provides an overview of the MbSE workflow. It shows for each of the SE phases the generated models together with the associated *Rhapsody* projects. How the different projects are managed from the configuration point of view will be detailed in section 4.

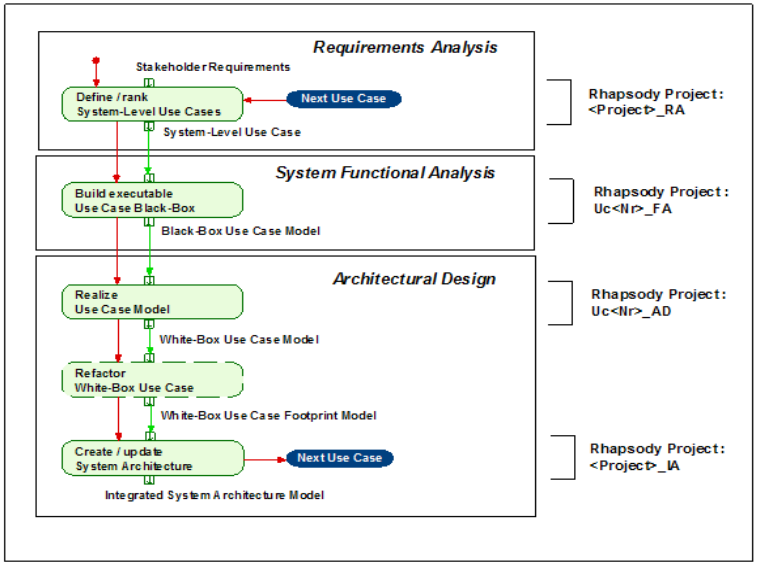


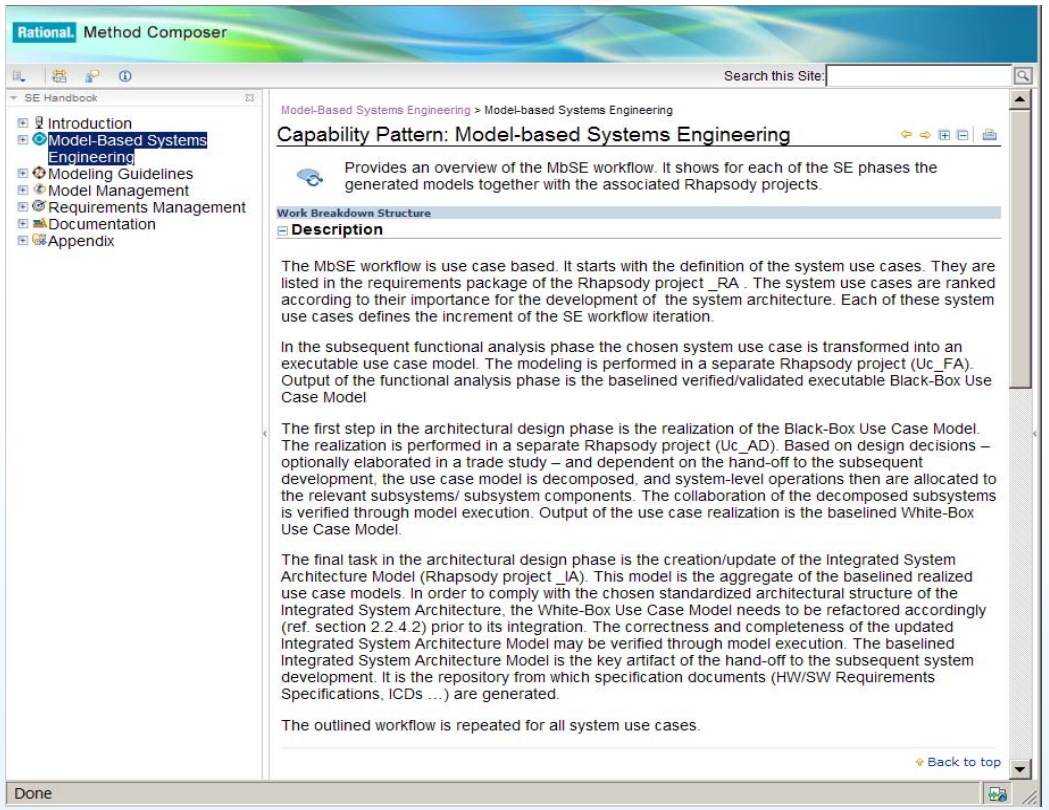
Fig. 2.2-1 Model-based Systems Engineering Workflow in the xxxx Project

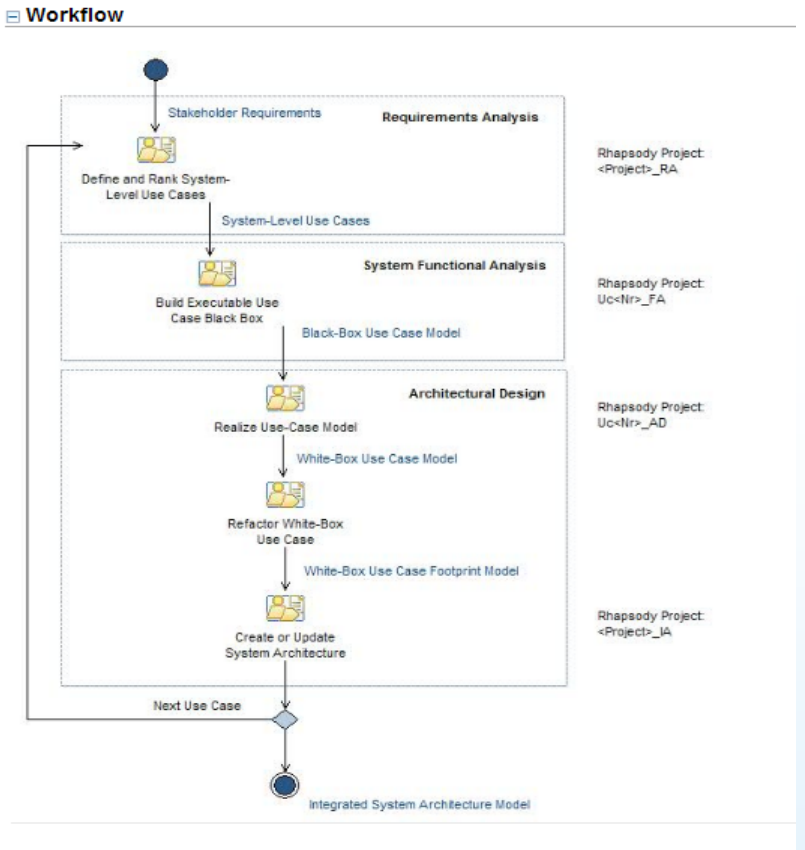
The MbSE workflow is use case based. It starts with the definition of the system use cases. They are listed in the requirements package of the *Rhapsody* project *<ProjecNamet>*_**RA** (ref. Section 4). The system use cases are ranked according to their importance for the development of

MbSE Handbook

Standardizing the MbSE workflow and Rhapsody tool usage

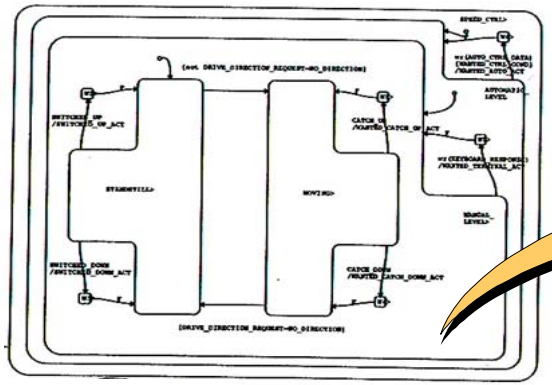
The Alternative: MbSE Handbook in RMC

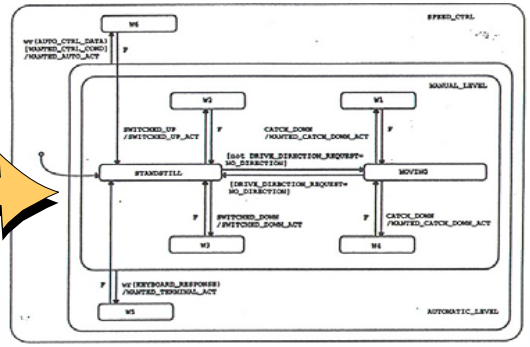


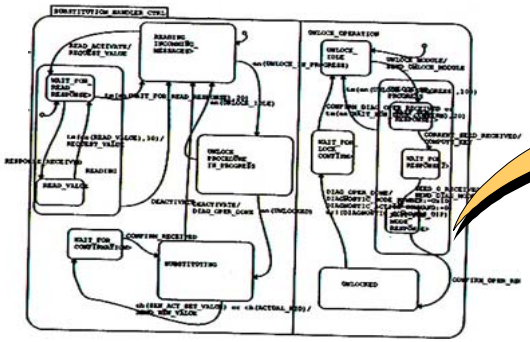


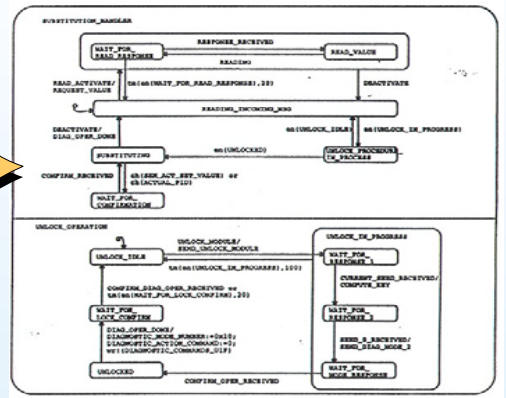
MbSE Handbook

Standardizing the usage of the modeling language



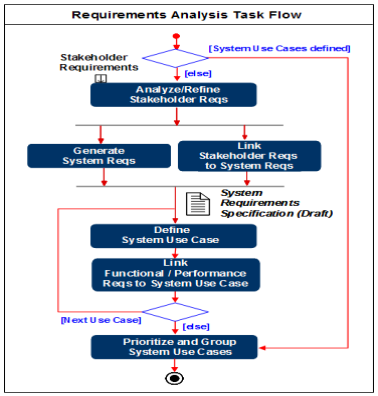


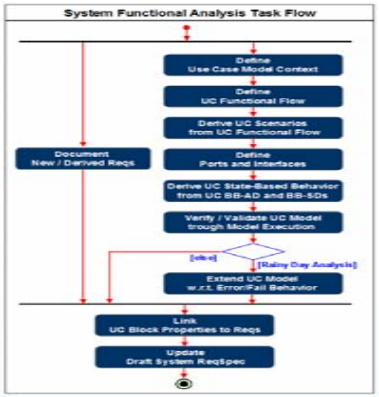


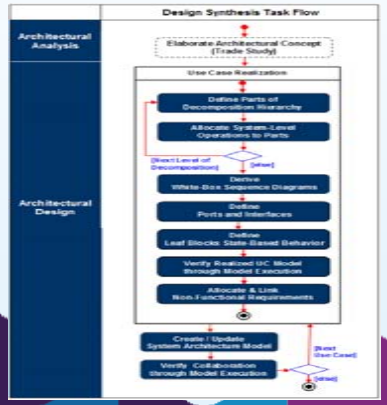


Deploying model-based systems engineering

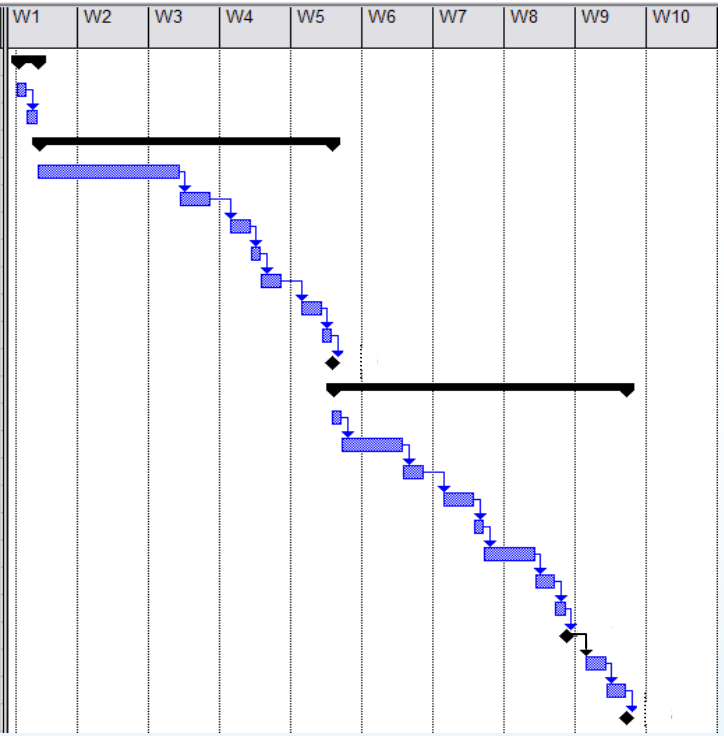
Managing the modeling activities: Project plan







	Task Name	Duration
1	**Requirements Analysis**	**2 days**
2	Define Use Case	1 day
3	Allocate Reqs to Use Case	1 day
4	**System Functional Analysis**	**21 days**
5	Create Black-Box Activity Diagram	10 days
6	Review & Update Black-Box Activity Diagram	3 days
7	Derive Use Case Black-Box Sequence Diagrams	2 days
8	Define System-Level Ports & Interfaces	1 day
9	Create Use Case Statechart	2 days
10	V&V Use Case Model through Model Execution	2 days
11	Link Model Properties to Requirements	1 day
12	Black-Box Use Case Model Baseline	0 days
13	**Architectural Design**	**21 days**
14	Define System Architectural Structure (BDD, IBD)	1 day
15	Create White-Box Activity Diagram	4 days
16	Review White-Box Activity Diagram	2 days
17	Derive Use Case White-Box Sequence Diagrams	3 days
18	Define System Architecture Ports & Interfaces	1 day
19	Create Subsystem Components Statechart	3 days
20	Verify Realized Use Case Model through Model Execution	2 days
21	Allocate & Link Non-functional Requirements	1 day
22	Realized Use Case Model Baseline	0 days
23	Generate Report	2 days
24	Merge Realized UC Model in Integrated System Architecture Model	2 days
25	Integrated System Architecture Model Baseline	0 days

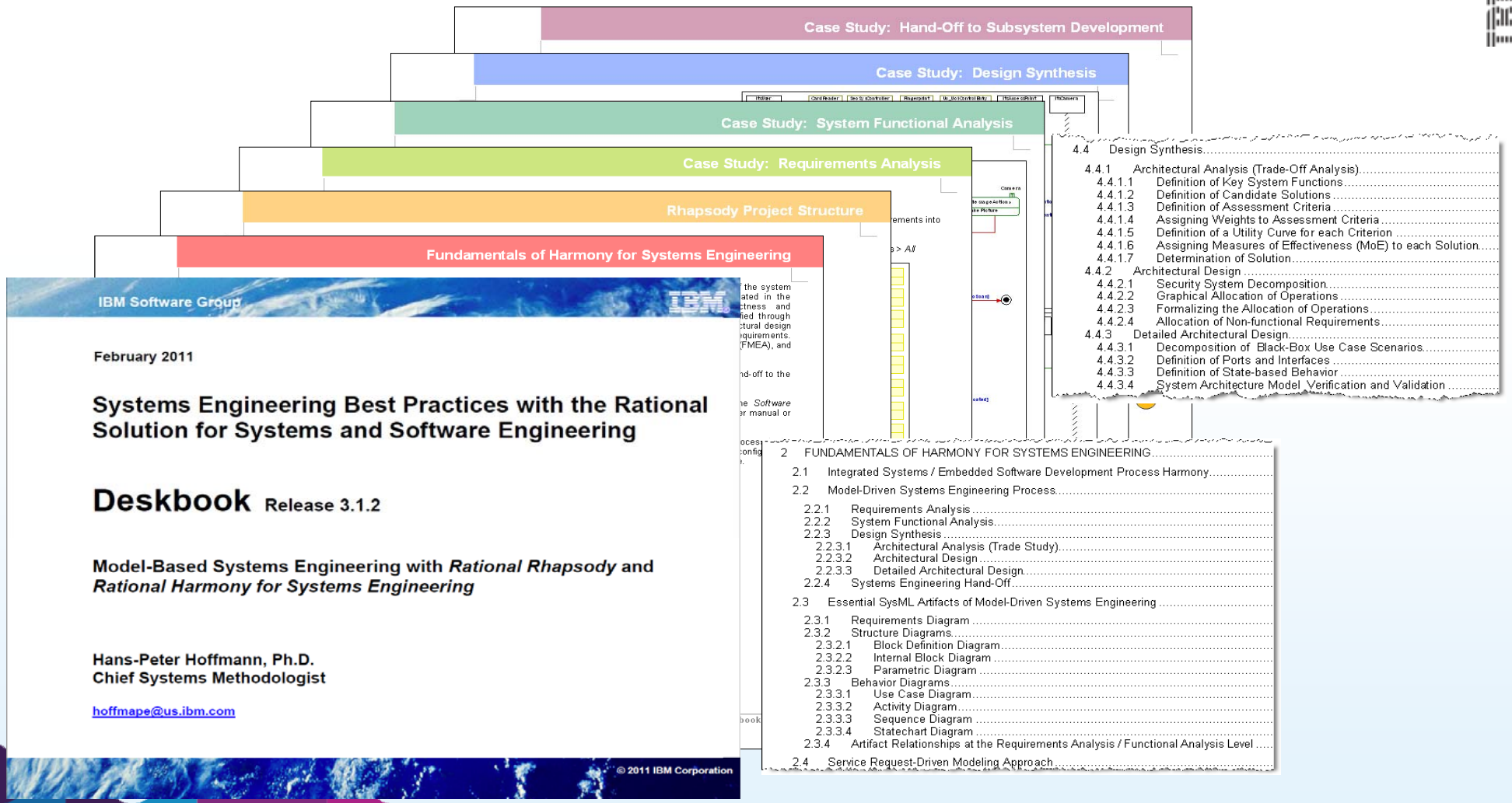


Agenda

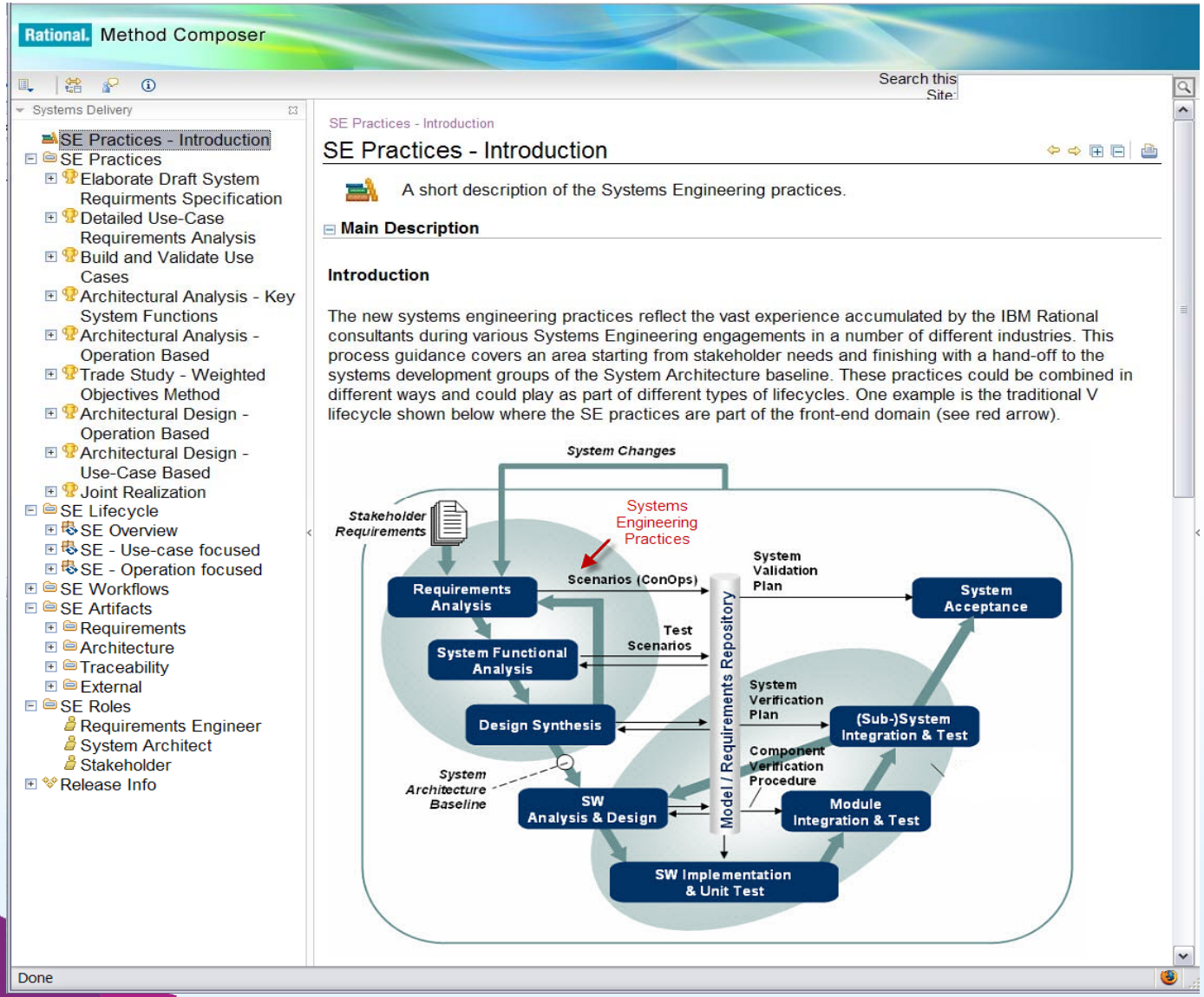
- Model-based systems engineering in a model-driven development lifecycle
- Essential SysML artifacts of the Rational MbSE approach
- Task flow in Rational *Harmony™ for Systems Engineering*
- Deploying MbSE with Rational® *Rhapsody®*
- **Documentation of Rational *Harmony™ for Systems Engineering***

Rational *Harmony™ for Systems Engineering* Deskbook

Rational Rhapsody® tool focused documentation

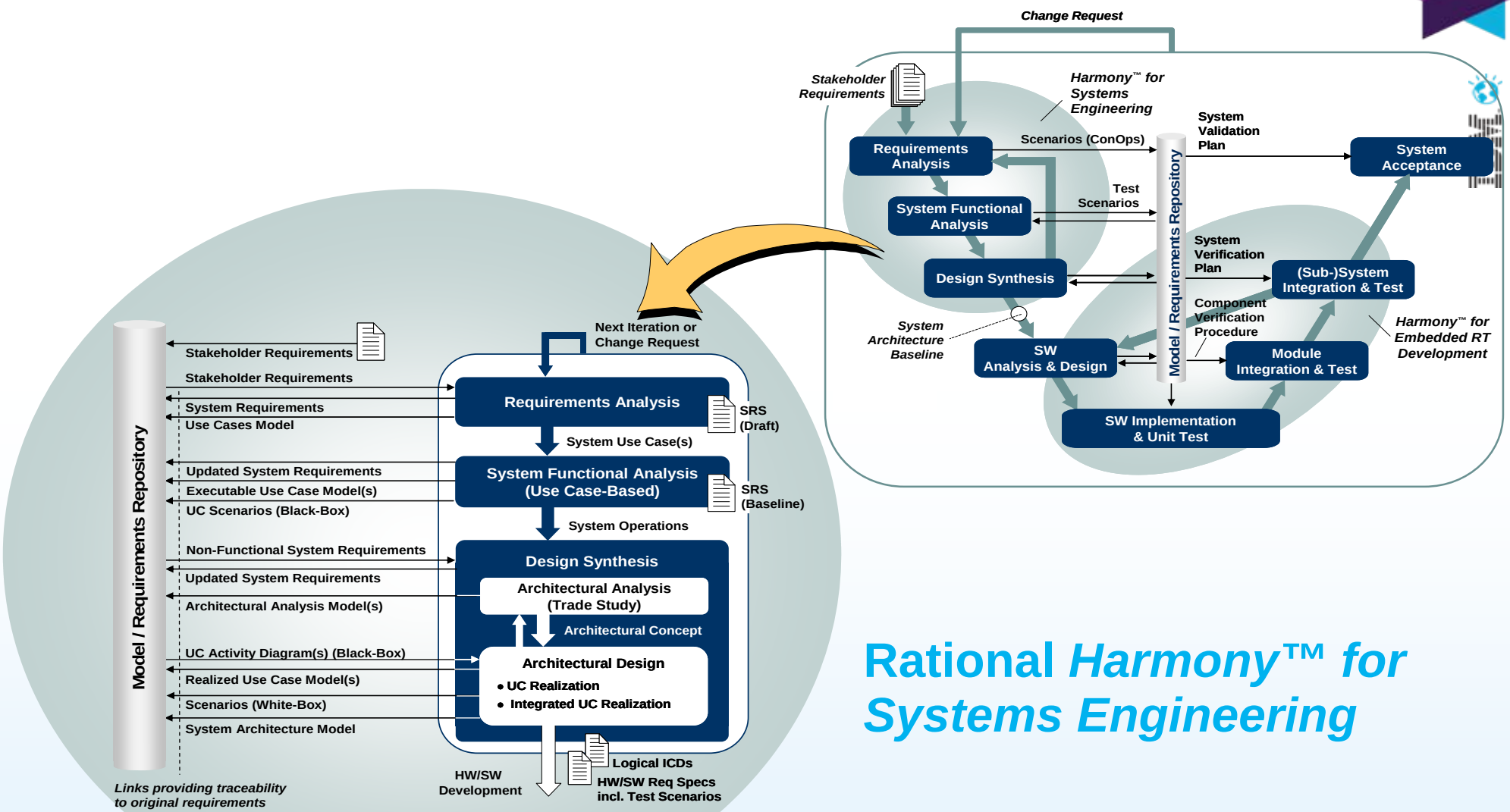


Rational Systems Engineering Practices captured in RMC



Why MbSE?
Experiences summary

- 60% Reduction in time to develop a specification
  - Due to improved clear up rate of issues
  - Improved communication
  - Models act as means of negotiating understanding between customer and supplie r
- Improved understanding of interfaces between contractors
  - Interfaces particularly complex as the boundary between the interfacing systems was not on the perceived physical boundary of the systems
- Improvements in HW/SW of 60% less errors
  - Due to better understanding of requirements
  - Tying the model and code together
- Use of a General Systems Architecture (library of reference models) has resulted in a reduction of 75% of the time taken to do initial bid work
  - 1 year to 3 months
  - Leads to reduced bid costs
  - Rapid response to bids

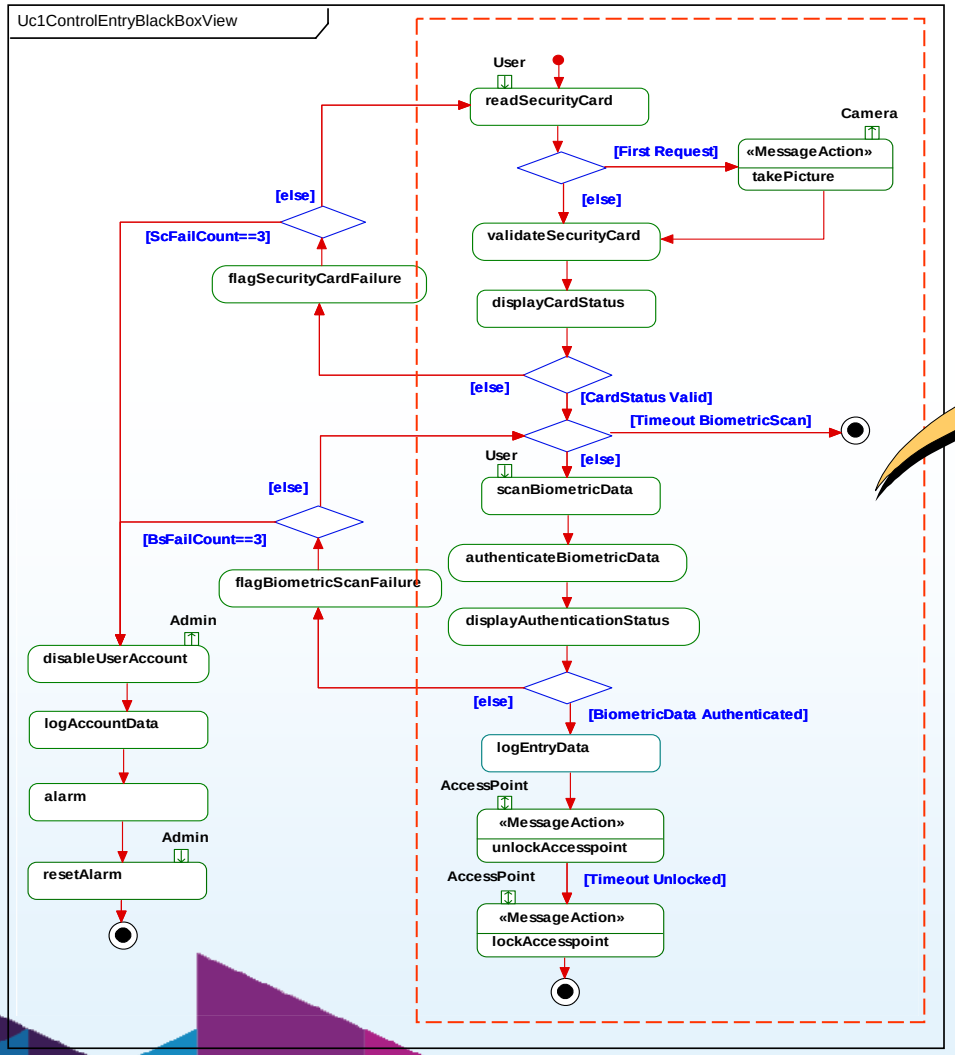


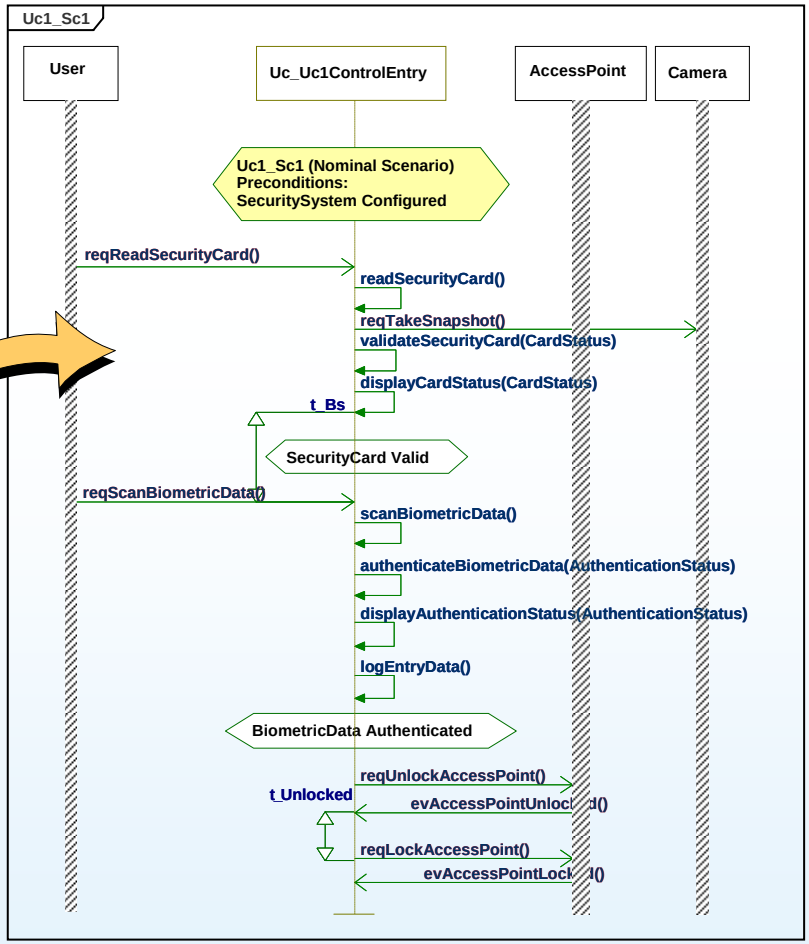
Rational *Harmony™ for Systems Engineering*

Backup

Case study

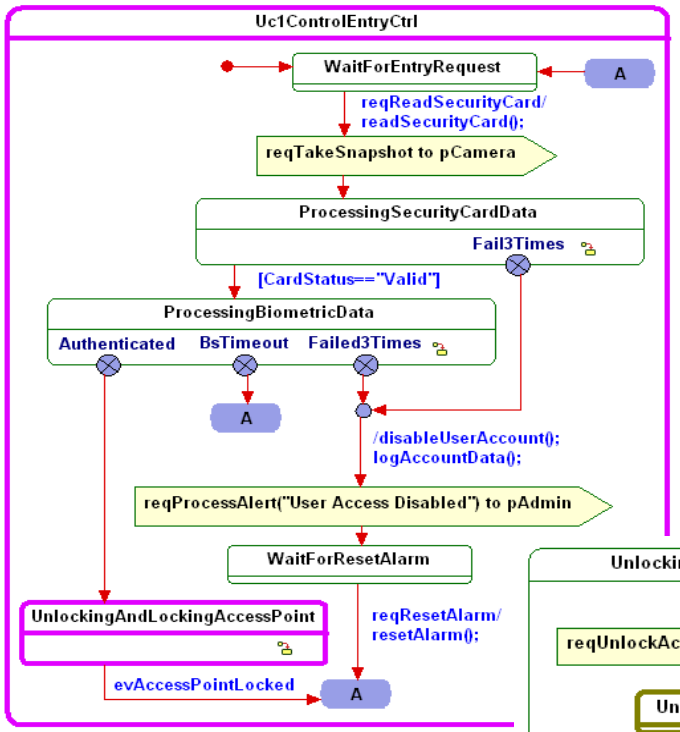
Derivation of a use case scenario from a use case black-box activity diagram



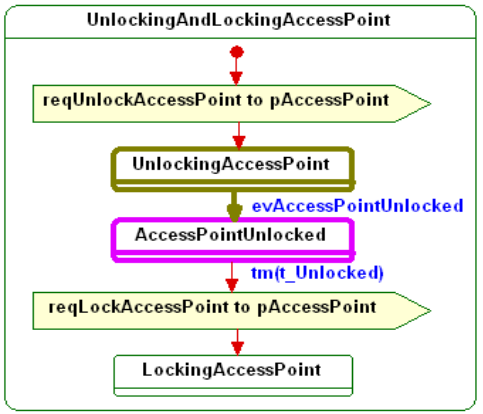


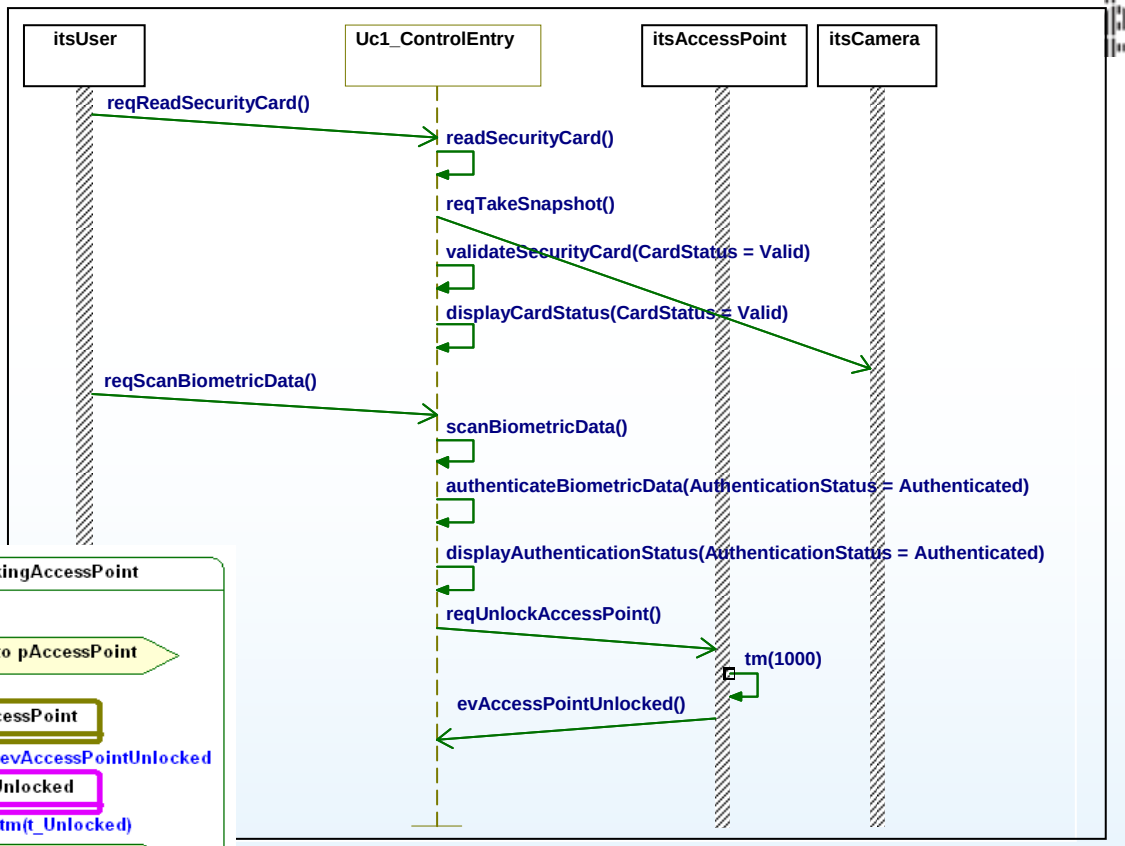
Case study

Model verification and validation through model execution (Rational *Rhapsody* tool)



Animated Statechart Diagram (Uc1ControlEntry)

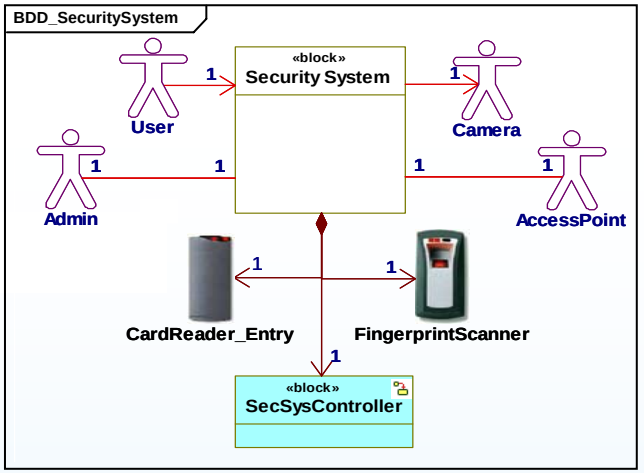




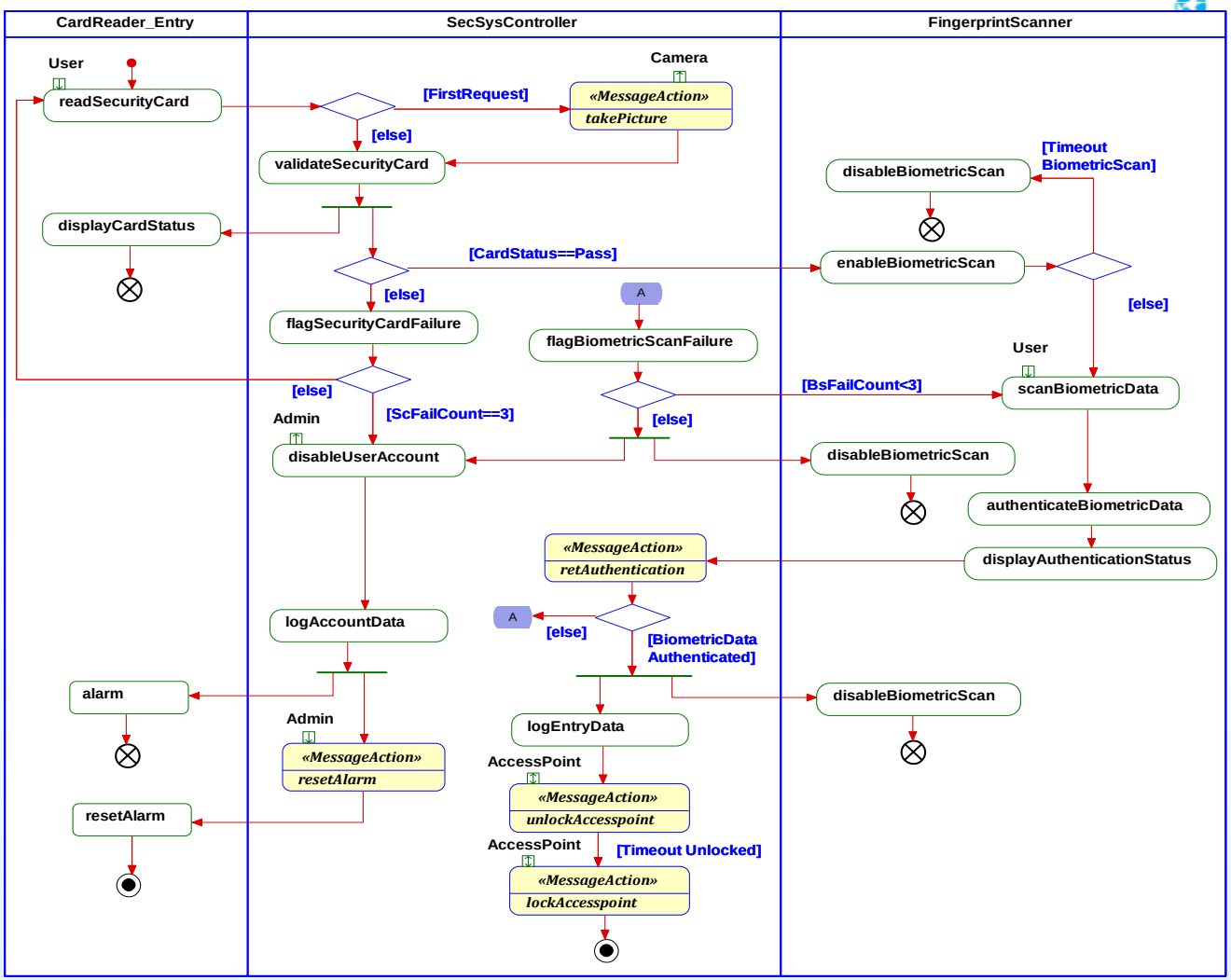
Animated Sequence Diagram (Uc1Sc1)

Case study – Uc1 (Control Entry) realization

Allocation of system-level operations to parts (subsystems)



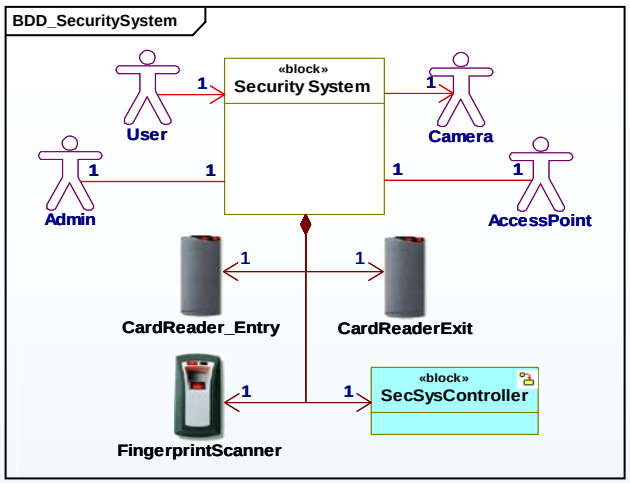
Chosen system architecture captured in a Block Definition Diagram

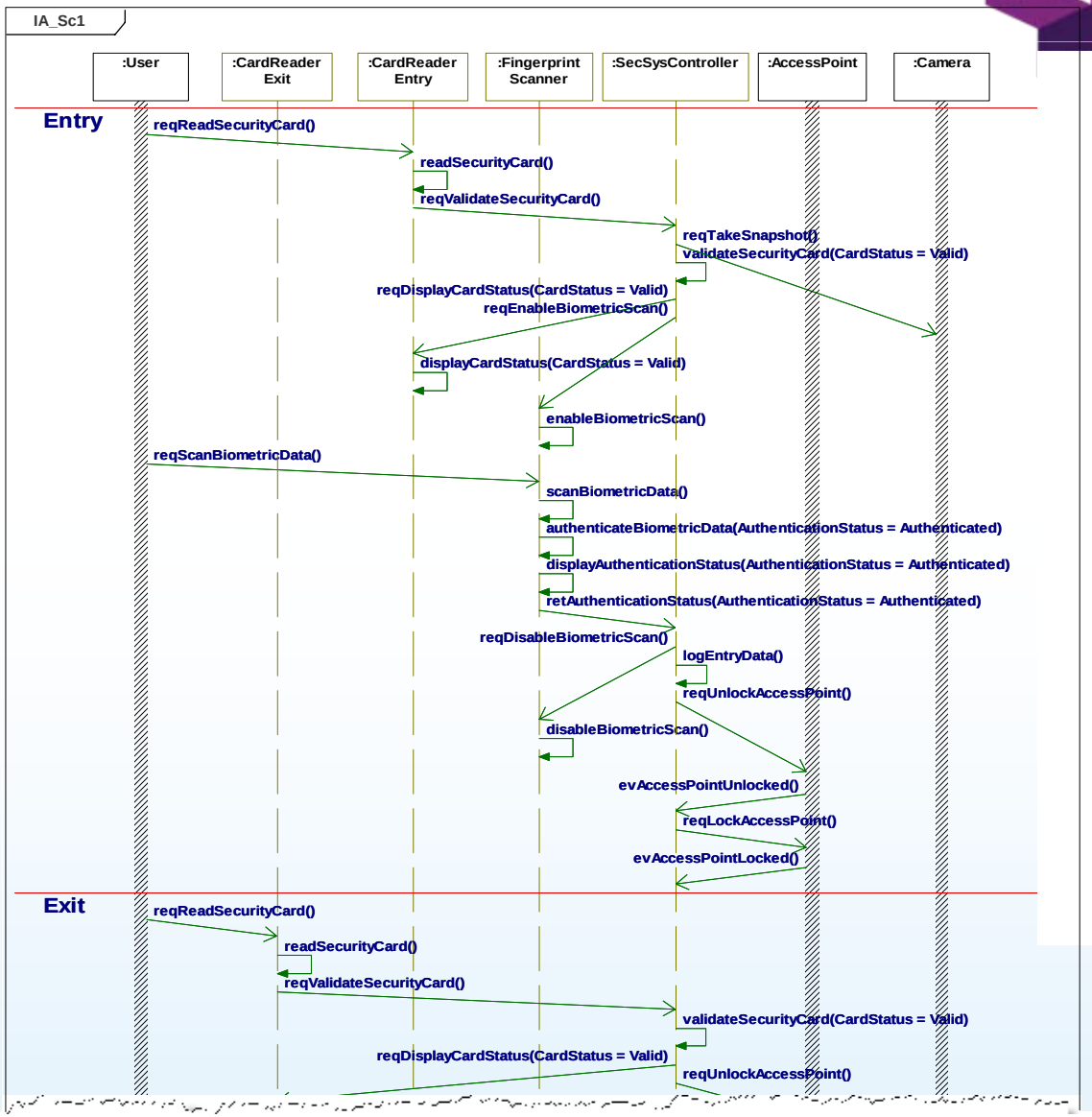


Uc1ControlEntry White-Box Activity Diagram

Case study

Integrated System Architecture verification through model execution





Animated Sequence Diagram (Uc1, Uc2 Nominal Flows)

www.ibm/software/rational

Software. Everyware.