

# Experiences using IBM Rational Software Modeler at Credit Suisse IT Private Banking

IBM Software Rational Developer Platform Event, Zürich

Date: 16.09.2009

Produced by: Karl Holik, Credit Suisse



# Personal Introduction: Karl Holik



*"I decide with which mood I go to work"*

## Personal data

- Age: 34 years
- Single
- Languages: German, English, French, Spanish, Czech

## Education

- March 2001: MSc ETH Physics

## Professional life

- Jun 01 – Aug 04: Accenture (IT Consultant)
- Sep 04 – Mar 07: Helsana (Chief of Staff IT, IT Project Manager, Program Manager)
- Since May 07: Credit Suisse
  - Head of Development Support / SW Architects DWH
  - Process Manager Solution Engineering
  - Process Manager Client Account Mgmt. / IT Product & Service Mgmt.

## My leisure time

- Travelling, Diving, Horse Riding, Hiking
- Dancing (Salsa), Gourmet stuff

# Agenda

1

**A brief Introduction** to Credit Suisse IT, the Credit Suisse IT Solution Delivery Standard Process and the Credit Suisse IT Standard Tool-Chain

2

**The Software Architecture Document (SAD)** – Next Generation of Software Modeling at Credit Suisse with Rational Software Modeler

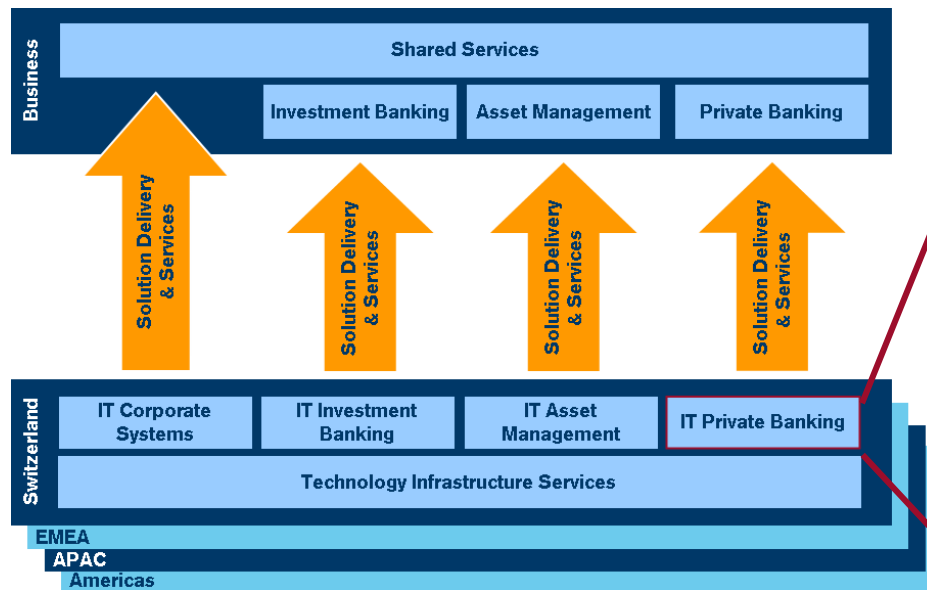
3

**An Outlook to the Future:** Rational Software Modeler at Credit Suisse IT

4

**Questions**

# 1. Credit Suisse IT Switzerland Organization and Key Figures

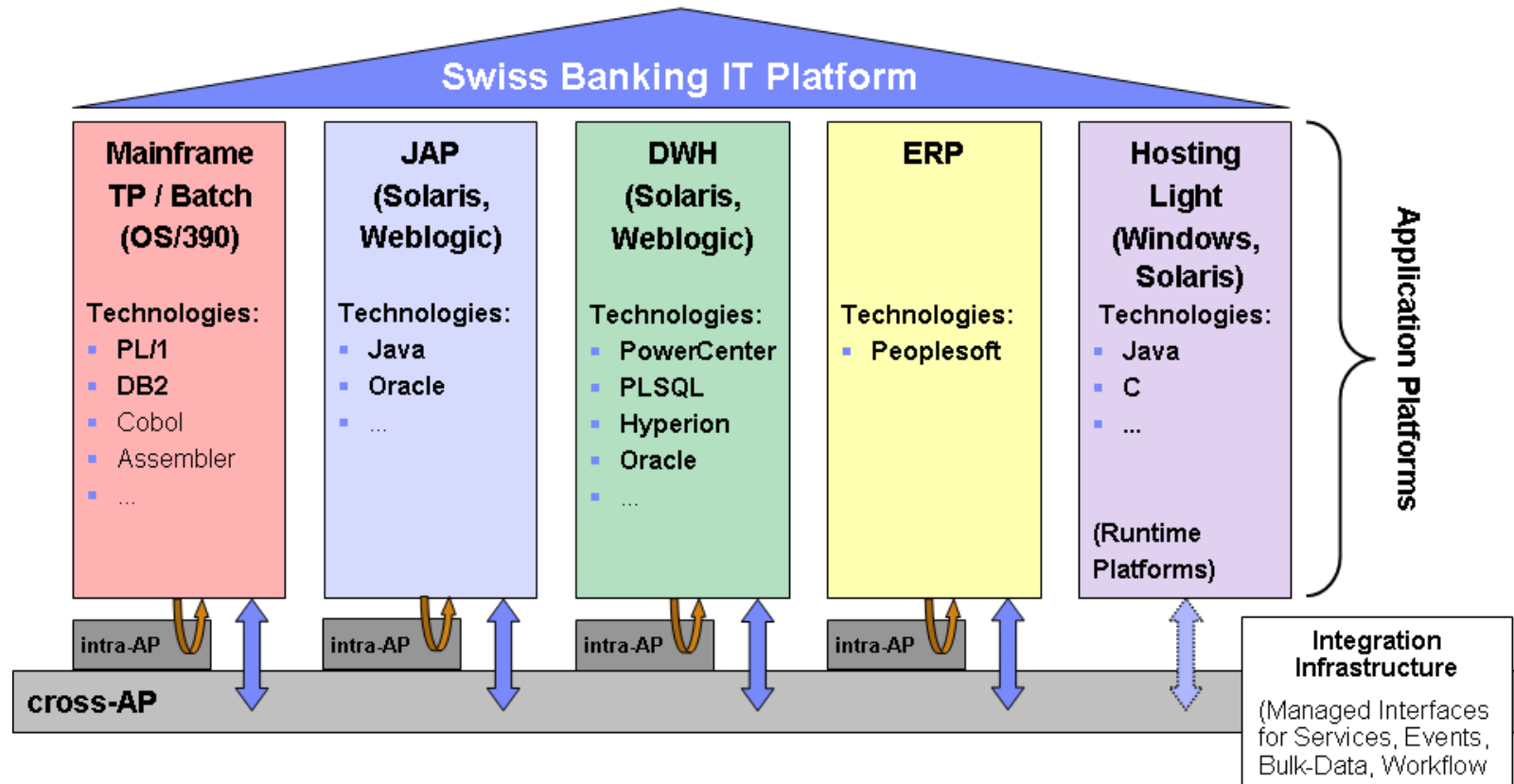


Credit Suisse IT PB	
Number of servers	6,750 Servers (Windows, UNIX, z/OS)
Number of applications	~ 800 applications (highly integrated)
Lines of Code	Java: ~ 11 Mio. PL/1: ~ 32 Mio.
Payment transactions	~ 250 Mio. / year
Printed pages	~ 224 Mio. / year
Emails	~ 339 Mio. / year
Employees	~ 4'000



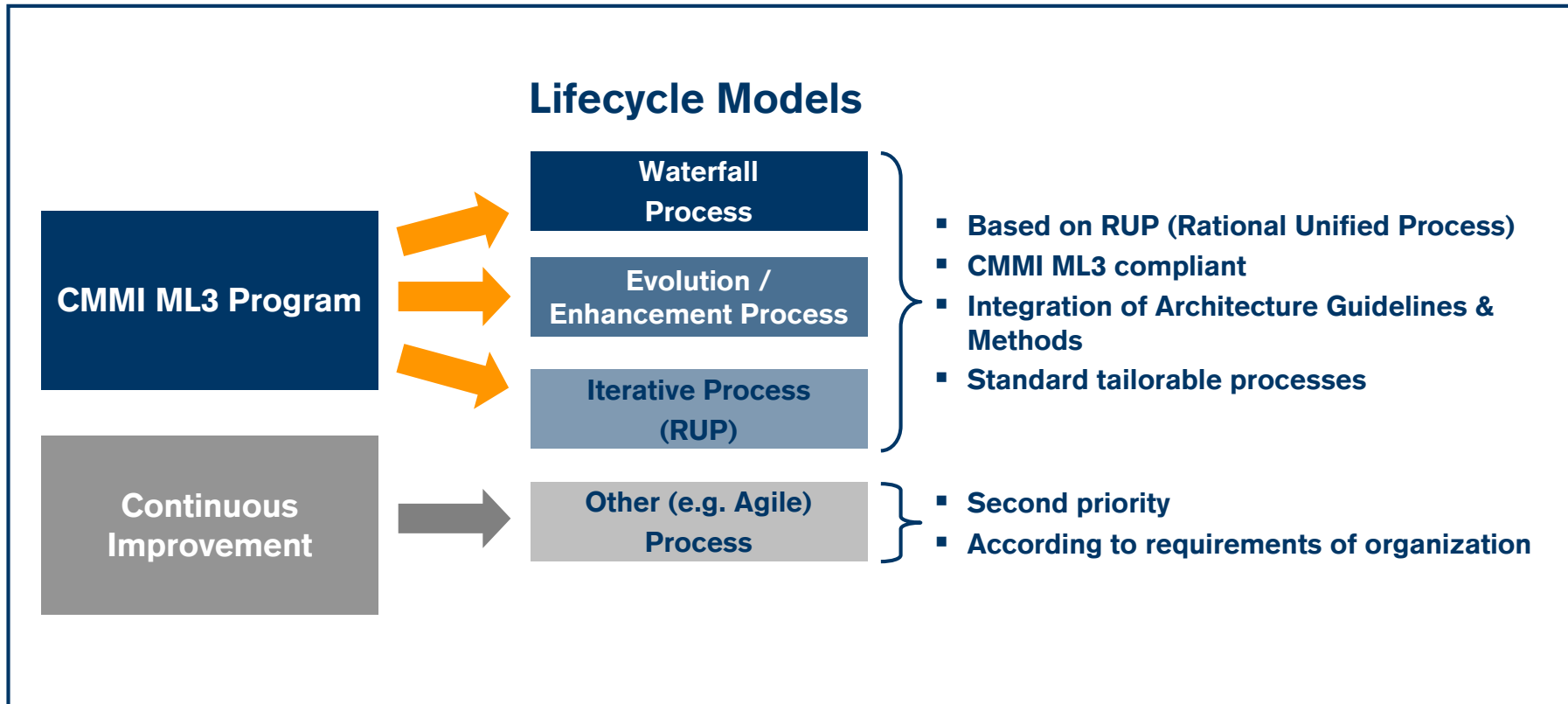
- Credit Suisse IT has one client, its Business → Banking (Private Banking, Investment Banking, ...) and Shared Services (Human Resources, Risk Management, Legal & Compliance, ...)

# 1. The Swiss Banking IT Platform (SBIP) – Technologies and Integration

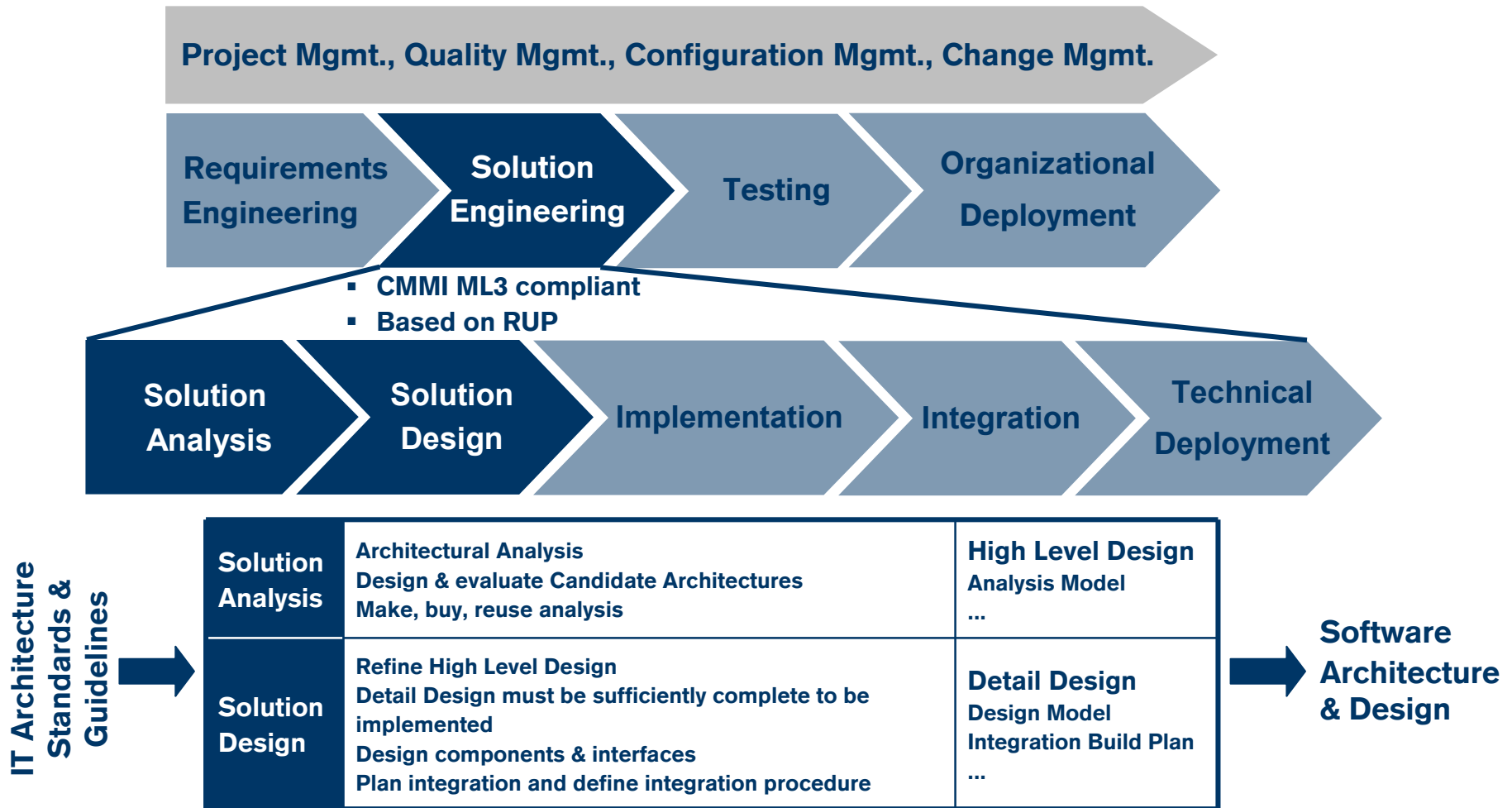


- Applications are deployed on 5 standardized Application Platforms
- The IT PB application landscape is highly integrated
- Most of applications run on Mainframe (PL/1) and JAP (Java)

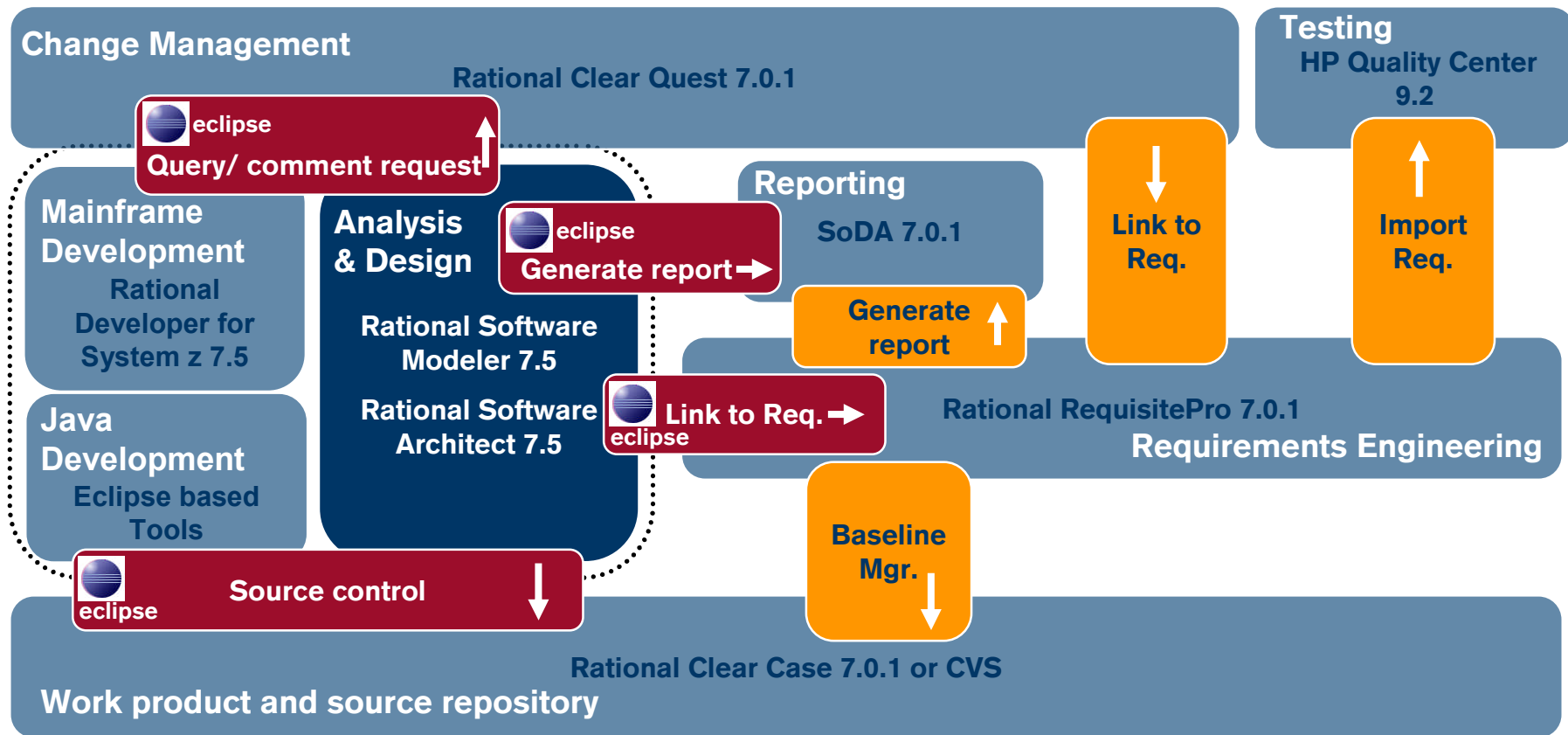
# 1. Development Process Lifecycle Models at Credit Suisse IT PB CH



# 1. Software Design in context of the Credit Suisse IT PB CH Solution Delivery Process



# 1. A Standard Tool Chain is one Element in bringing Processes alive and making Processes practicable



**Note:** Arrow directions document integration dependencies, not necessarily navigation/information flow.



# Agenda

1

**A brief Introduction** to Credit Suisse IT, the Credit Suisse IT Solution Delivery Standard Process and the Credit Suisse IT Standard Tool-Chain

2

**The Software Architecture Document (SAD)** – Next Generation of Software Modeling at Credit Suisse with Rational Software Modeler

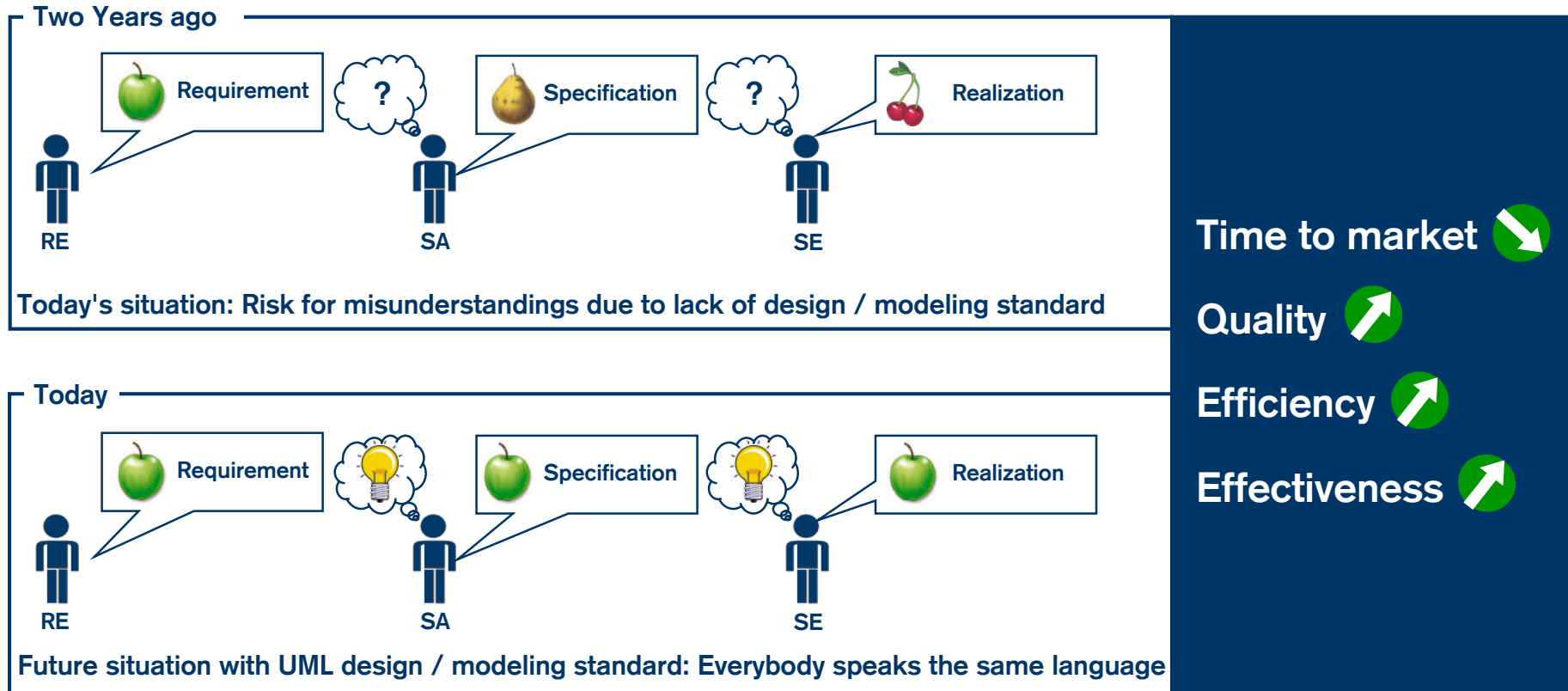
3

**An Outlook to the Future:** Rational Software Modeler at Credit Suisse IT

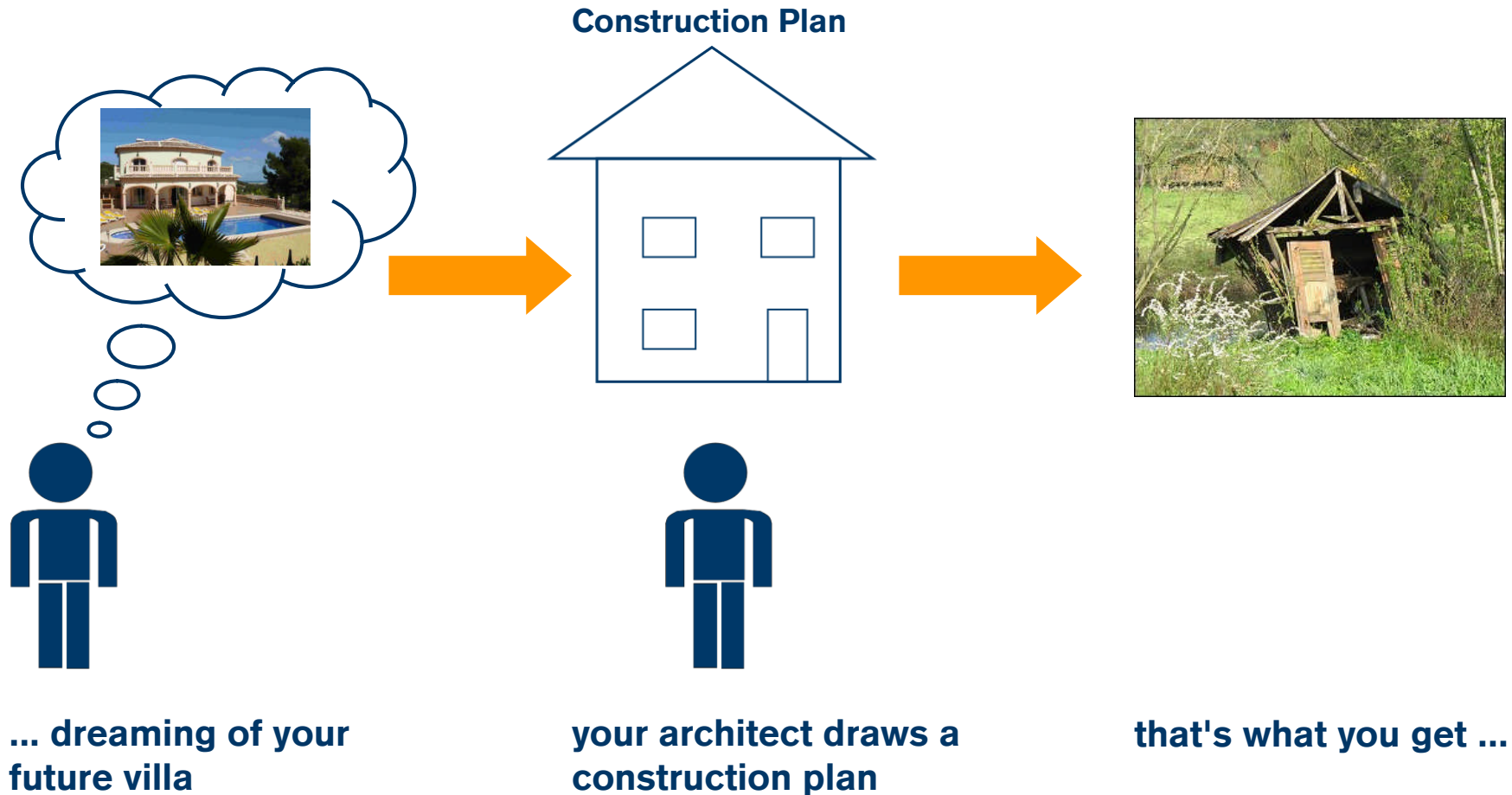
4

**Questions**

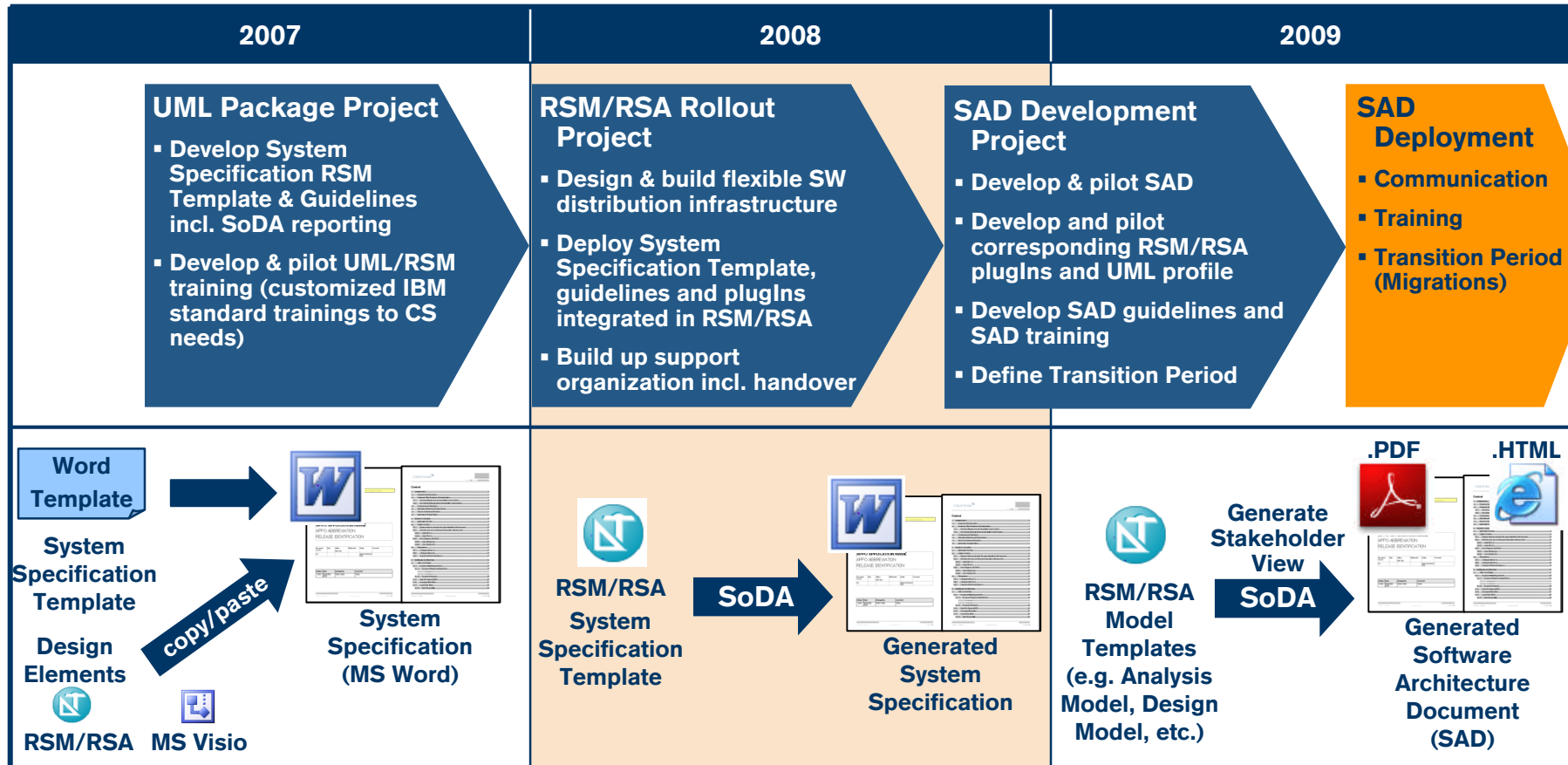
## 2. The Case for UML



## 2. The Case for the Software Architecture Document (SAD)



## 2. Software Architecture: The way from the paper-based System Specification to the model-centric Software Architecture Document (SAD)



Abbreviations: RSM = Rational Software Modeler, RSA = Rational Software Architect, SoDA = Rational SoDA

## 2. Motivation for the Software Architecture Document: Pains and Common Questions regarding the System Specification from our IT Community

### Feedback from our Software Engineers and Software Architects community:

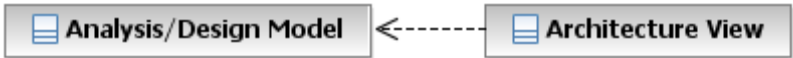
- "I don't know where to find **current documentation for my application** – do I have to create a full system specification for this little change in my project?"
- "Why do I have to include the **same information in several documents** for review purposes?"
- "Why is there **no common software engineering / visual modeling tool** for specification and documentation?"
- "Using UML for software design is only suitable for object oriented programming languages such as Java and not for structured programming such as **PL/1. I want to stay with Jackson!**"

## 2. Purpose of the Software Architecture Document (SAD)

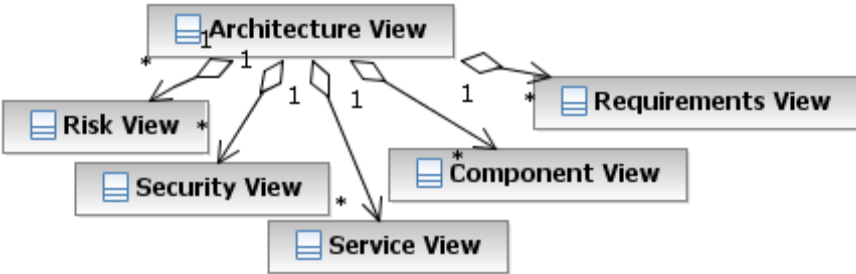
- Record the **architectural and design** decisions for an application
  - ▶ structure: components, data models (logical/physical)
  - ▶ behavior: interfaces/services required & implemented, user interface concept/design
  - ▶ deployment: technical architecture (platform, nodes)
  
- Focus on **architecturally significant requirements**
  - ▶ essential functionality
  - ▶ non-functional requirements such as quality, reliability, performance, security, maintainability, configurability etc.
  
- It is a **product document** (1 application = 1 SAD)
  - ▶ required document as per IT PV milestones (initialization, system design, deployment)
  - ▶ *not* part of project documentation (e.g. project plan)

## 2. Goals of the Software Architecture Document (SAD)

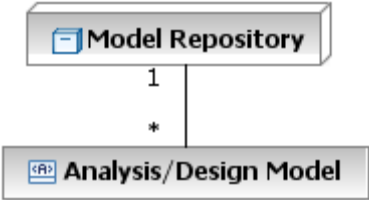
- 1 Software Architecture Document is a **View** to an application's model, not a document




```
graph LR; AV[Architecture View] -.-> ADM[Analysis/Design Model]
```
- 2 Stakeholder views can be specified and independently customized/adjusted to a particular stakeholder's requirement



```
graph TD; AV[Architecture View] -- 1 --> RV[Risk View]; AV -- 1 --> SV[Security View]; AV -- 1 --> SSV[Service View]; AV -- 1 --> ReqV[Requirements View]; RV -- * --> SSV; SV -- * --> SSV; ReqV -- * --> CV[Component View]; CV -- * --> SSV
```
- 3 A repository stores multiple revisions of an application's model, it's components, services and deployment nodes (**documentation modules**)

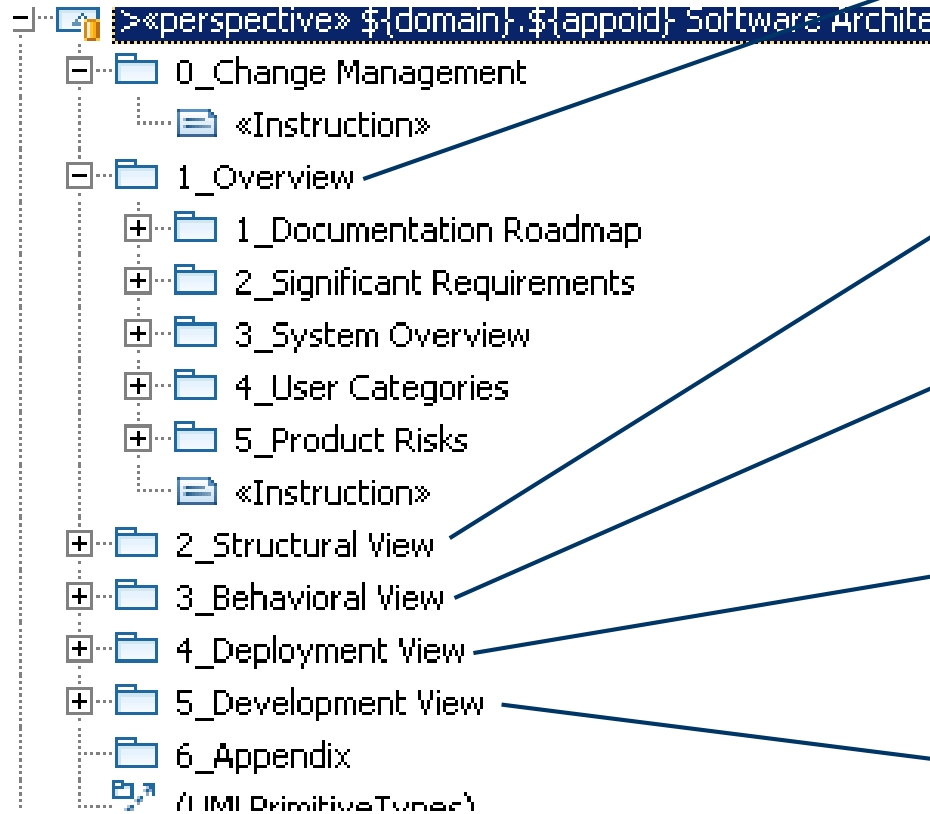


```
graph TD; MR[Model Repository] -- 1 --> ADM[Analysis/Design Model]; ADM -- * --> MR
```
- 4 Releases are managed using configuration management (design model = configuration item)



```
graph LR; RP[Release Package] -.-> ADM[Analysis/Design Model]
```

## 2. Multiple Views – Separate Concerns



### Overview:

Introduction and placement in application landscape

### Structural View:

Which components and data make up the application?

### Behavioral View:

How do the components work together to realize the use cases?

### Deployment View:

Where are the components deployed, on which platforms?

### Development View:

Description of development environment



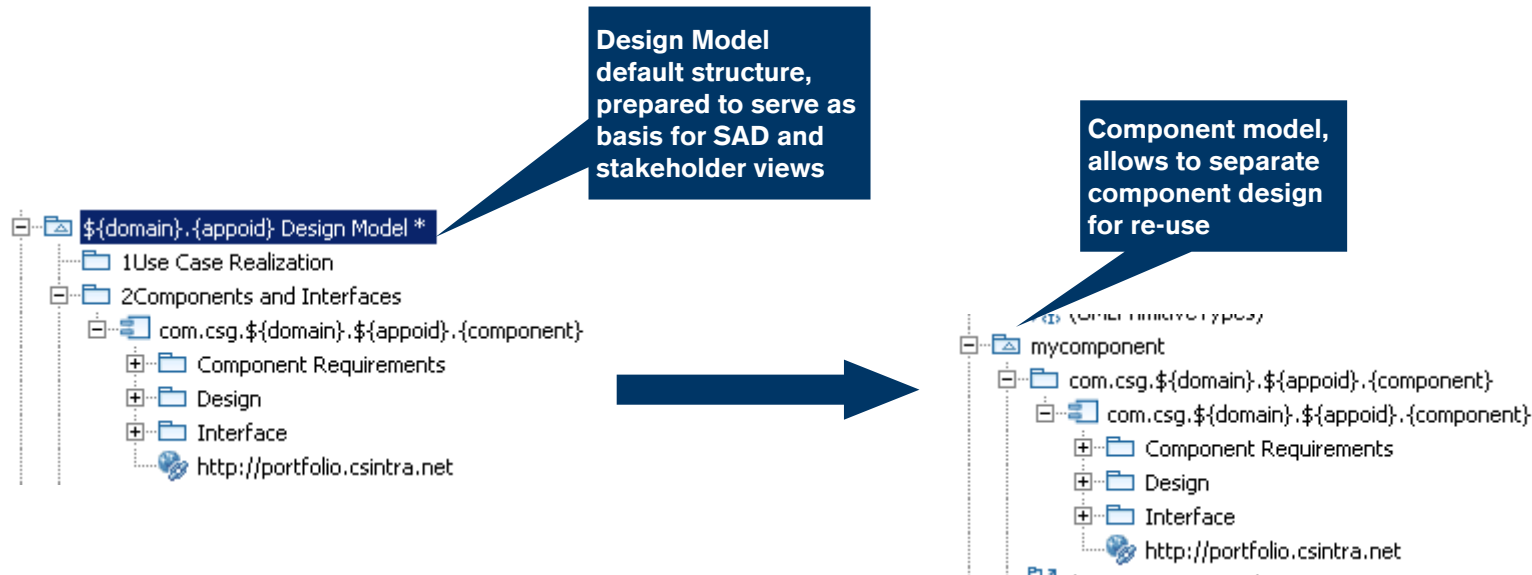
## 2. Design follows Credit Suisse Application Structure

### ▪ Application Design Model

Application-level design model contains the full design of an application, including detail design specifications

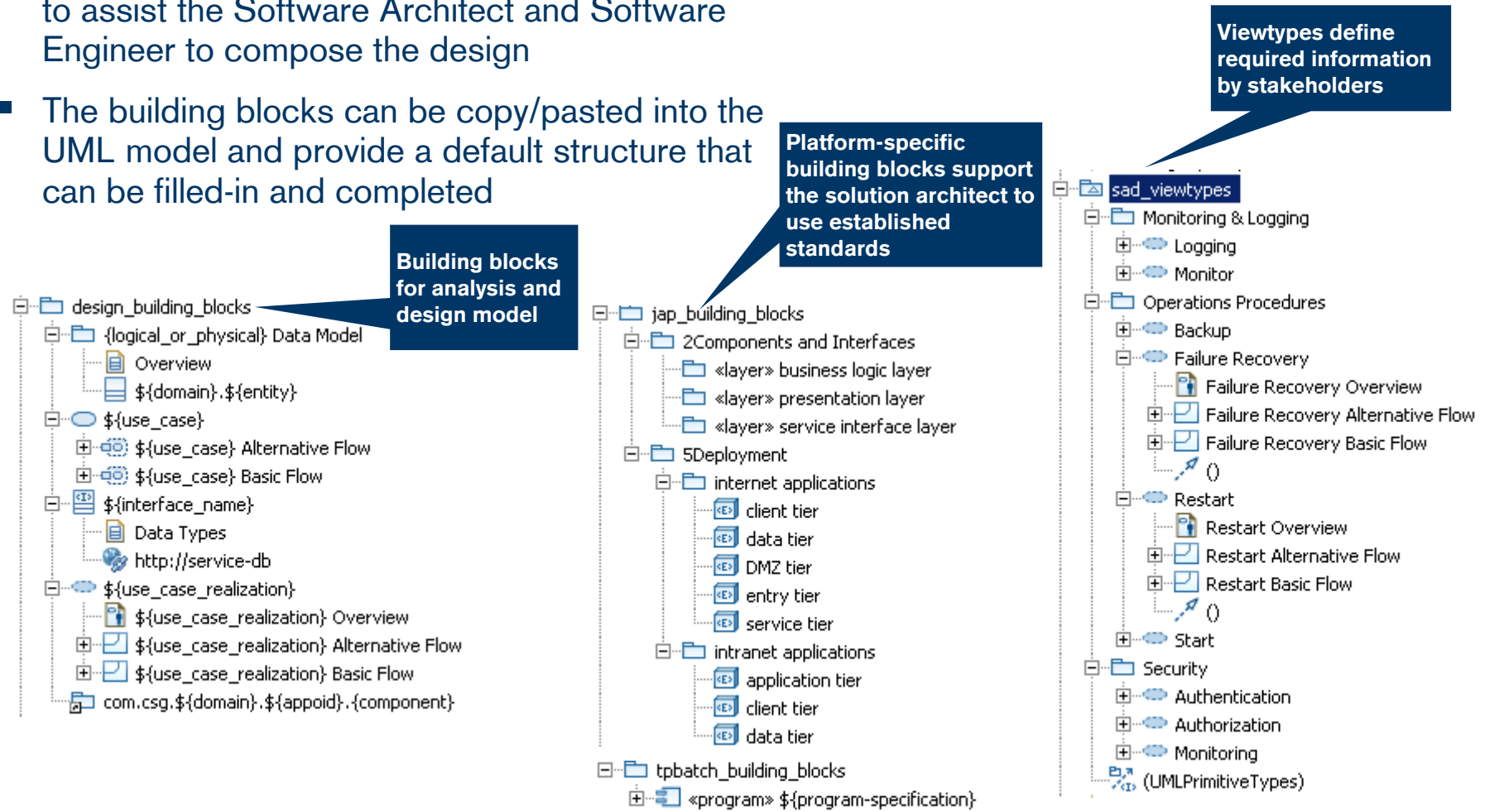
### ▪ Component Model

Every component is detailed in a component specification, kept integrated within the design model, or extracted into its own model file for better handling and re-use



## 2. Common Building Blocks speed up Design Tasks

- A set of "building blocks" are made available to assist the Software Architect and Software Engineer to compose the design
- The building blocks can be copy/pasted into the UML model and provide a default structure that can be filled-in and completed



## 2. Example of Platform-specific Building Blocks

Example for PL/1 (structured programming)

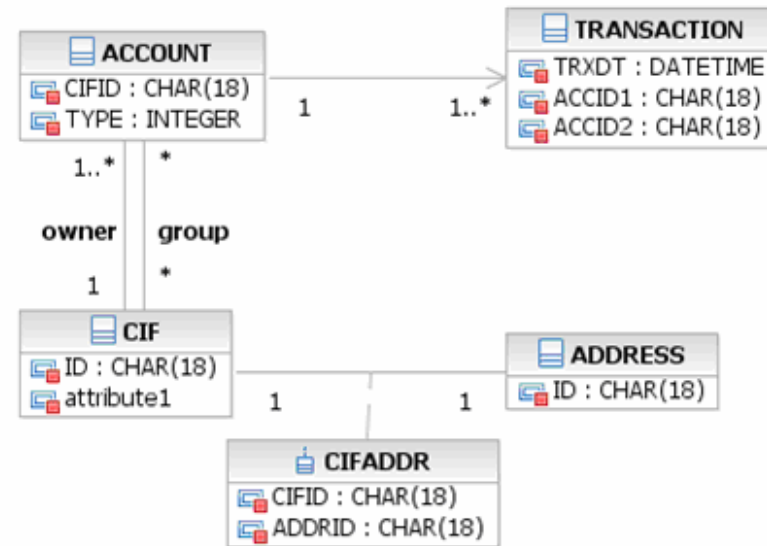
- A PL/I module for example, can be represented as a UML component, only exhibiting one operation and no data elements.
- A PL/I data structure can be represented likewise as a class, but only showing the data elements.

```
ACTRX: PROC (DTRXY);
DCL 1 DTRXY,
2 DATE CHAR (12)
2 AMOUNT BIN FIXED (32) ;
...
END ACTRX;
```

↓ PL/1 source code representation in UML as it will look like in the SAD

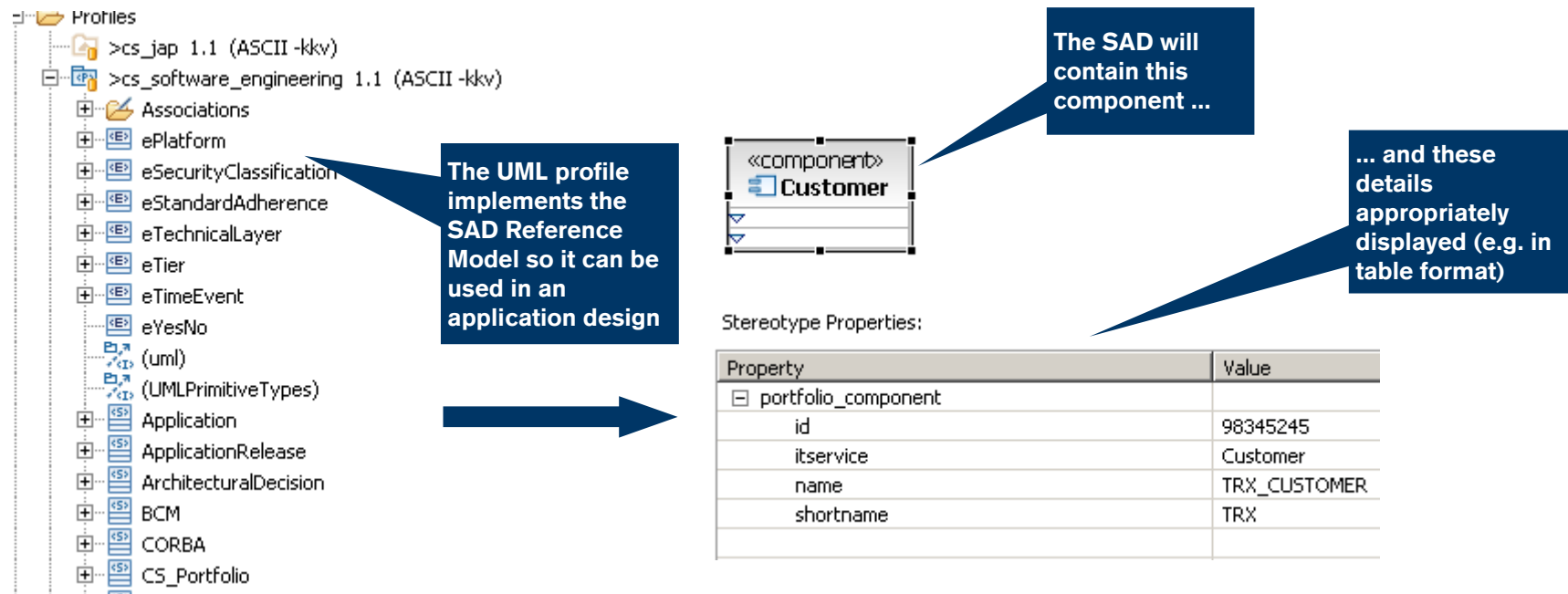


Example for a Data Model (Entity-Relationship-Diagram style)



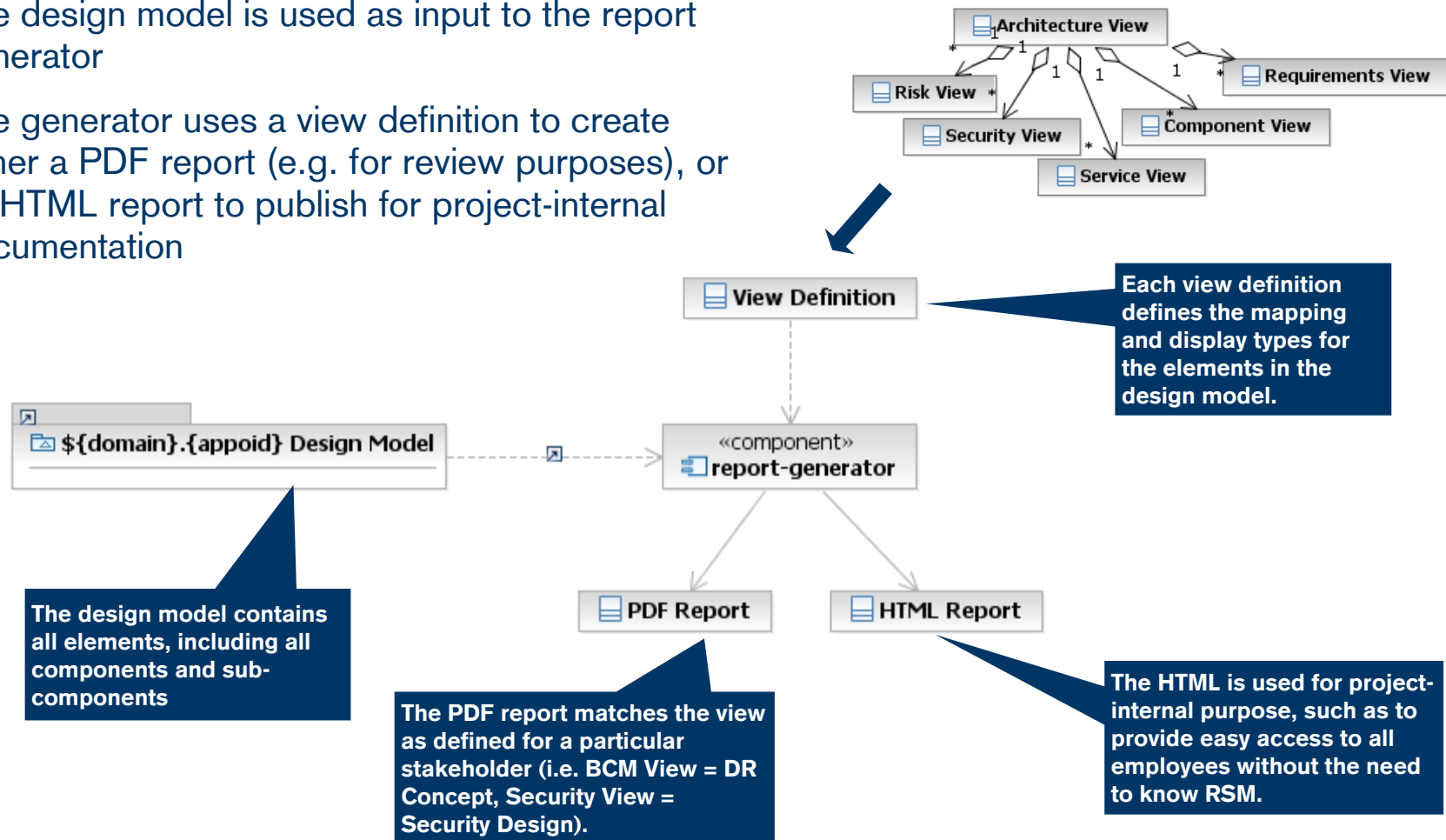
## 2. Common Design Elements: Capture detailed Information per Design Element

- Common design elements have pre-defined attributes.
- Attributes define the information that must/can be recorded (e.g. for an application, this would include the application domain name (application clustering), the unique CS application ID etc.; for a technical component it would include the application platform it depends on, the technical name etc.)



## 2. Software Architecture Document: View Definition and Generation

- The design model is used as input to the report generator
- The generator uses a view definition to create either a PDF report (e.g. for review purposes), or an HTML report to publish for project-internal documentation



## 2. Templates, Guidelines and Views integrated with Rational Software Modeler

- **Deliverables will be importable to RSM**
- **Guidelines will include:**
  - Setup modeling environment
  - Collaborate in team
  - Manage multiple releases
  - Define System Context
  - Define component model
  - Define service model
  - Define deployment model
  - Establish Traceability
  - Publish a SAD
  - Use RSM with SoDA
  - Define UI Concept & Model
  - Define PIM Service Model

SAD Process Asset (Deliverables)

<R> cs\_rsm\_checklists

Guidelines (PDF) and checklists (Eclipse cheatsheets)

<R> cs\_sad\_template

RSM Model Template and UML Profile

<R> uml\_modeling\_guideline

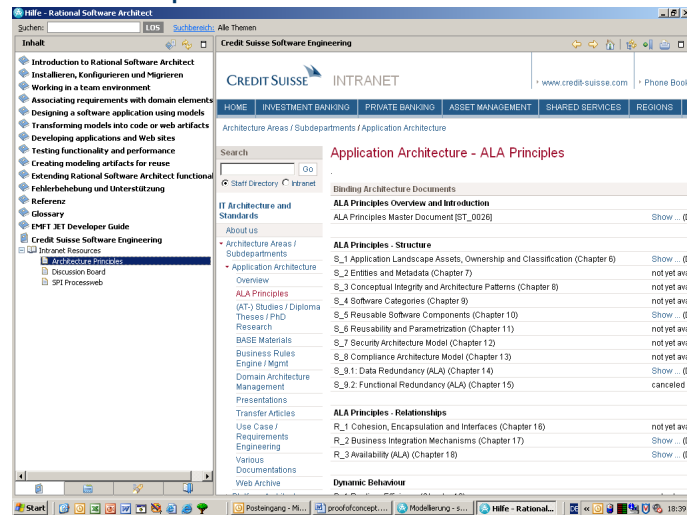
UML Modeling Guideline (update)

<R> soda\_view\_templates

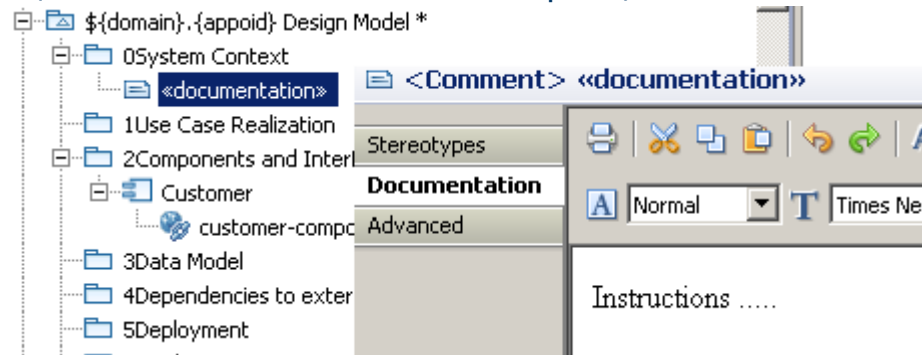
SoDA Templates for: SAD (including Security Design), IT DR Concept, IT Operations, Risk Profile

## 2. User Guidance and Reference Material available in Rational Software Modeler

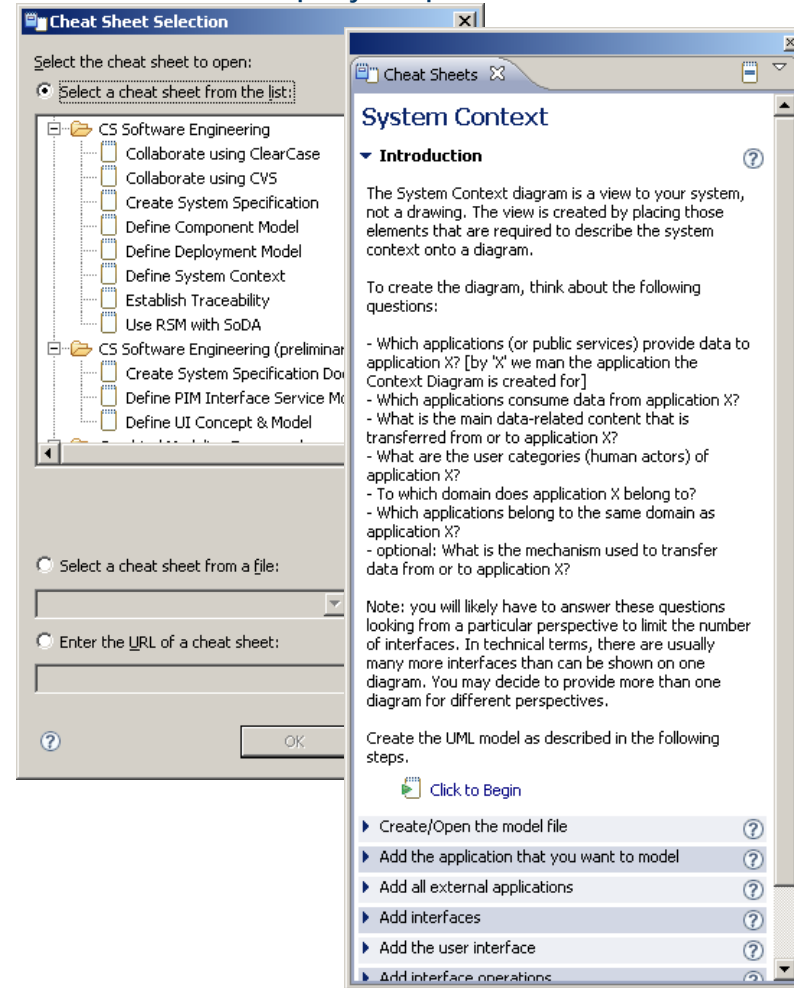
### 1 RSM Help links to Credit Suisse internal standard



### 2 Instructions within the UML model itself (same as comments in word template)



### 3 Checklists for step-by-step instructions



## 2. The Software Architecture Document (SAD) in a Complex Environment – Challenges

### Migration to the Software Architecture Document

- **Multiple System Specifications: inconsistent documents in various repositories**

→ **How to migrate from System Specification to SAD?**

### Parallel Development regarding SAD

- **Configuration management and splitting the model is the solution in theory**
- **But how does configuration management concretely work for versioning of packages, classes, etc.?**
- **How do we synchronize the SAD in the event of parallel development?**
- **How do we handle SAD in the context of Global Distributed Development (GDD)?**



## 2. Software Architecture Document: The Summary

- The Software Architecture Document is **model-centric**, is held **under configuration management** (like source code) and is an integral part of the application documentation.
- The implementation in Rational Software Modeler provides a **professional working-environment with out-of-the box integrations** to the Credit Suisse Standard Tool-Chain (e.g. Rational RequisitePro, Rational Clear Case, etc.).  
→ Be compliant to standard processes (CMMI ML3) without thinking about it.
- The **pre-defined building blocks (stereotypes with constraints)** provide a **minimal standard** for documenting software architecture at Credit Suisse IT, but do not exaggerate modeling standardization (potential reason for failure due to change resistance by senior software architects).
- The **automated generation of stakeholder reports** at any time and the predefined SAD template in Rational Software Modeler increase staff motivation → Work on content instead of setting up the work environment.
- Guidelines, links to Credit Suisse internal standards and step-by-step instructions will be delivered by the **Credit Suisse standard installation of Rational Software Modeler** to the end-users. → No individual customization and installation needed.

## 2. Software Architecture Document: The **real** Summary From Mud-Wrestling to a Controlled Ride



**Without Software Architecture Document (SAD)**

### **With SAD**

- Managed application architecture under CM
- Documentation corresponds to actual implementation
- Professional working-environment with integration & automation



## 2. Software Architecture Document: Mountains of Paper for Documenting Applications are History



# Agenda

1

**A brief Introduction** to Credit Suisse IT, the Credit Suisse IT Solution Delivery Standard Process and the Credit Suisse IT Standard Tool-Chain

2

**The Software Architecture Document (SAD)** – Next Generation of Software Modeling at Credit Suisse with Rational Software Modeler

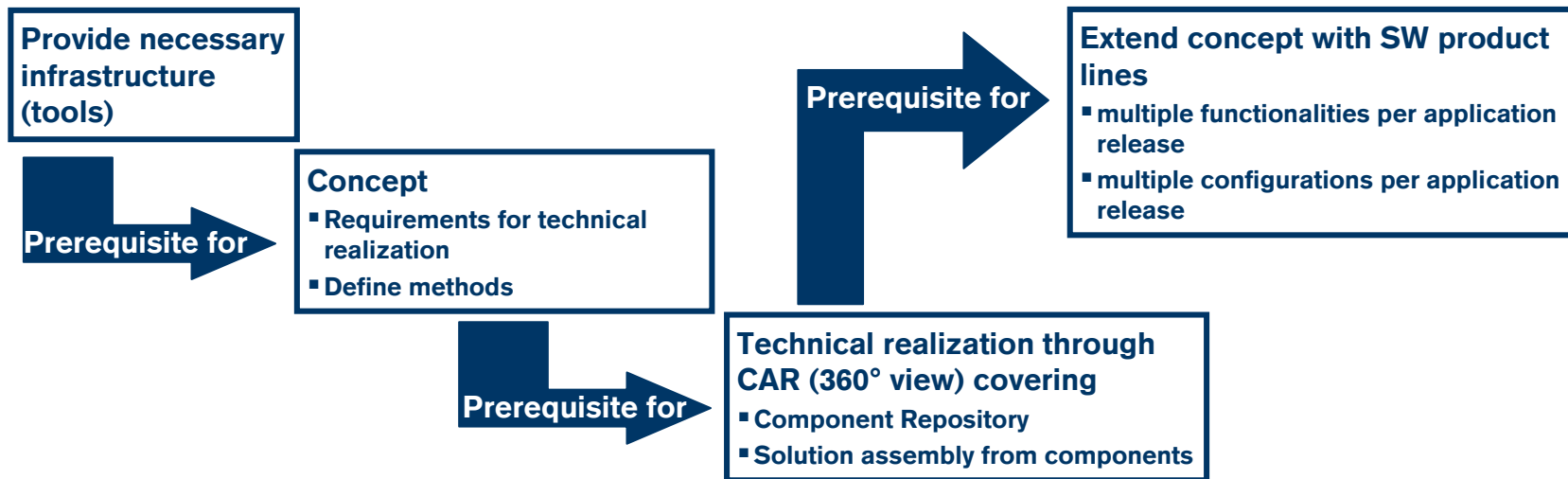
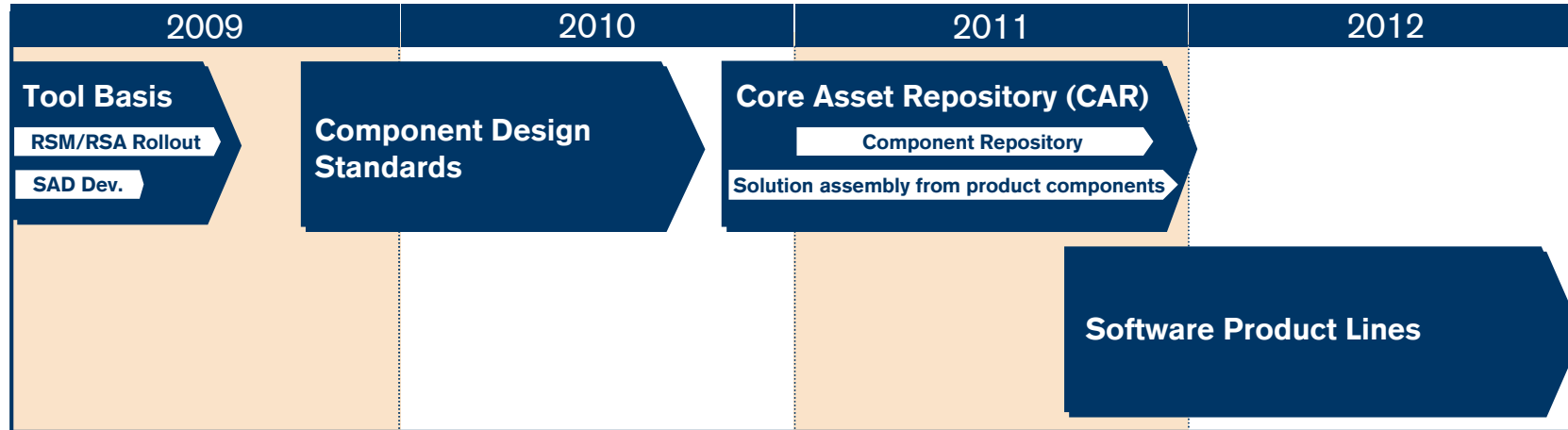
3

**An Outlook to the Future:** Rational Software Modeler at Credit Suisse IT

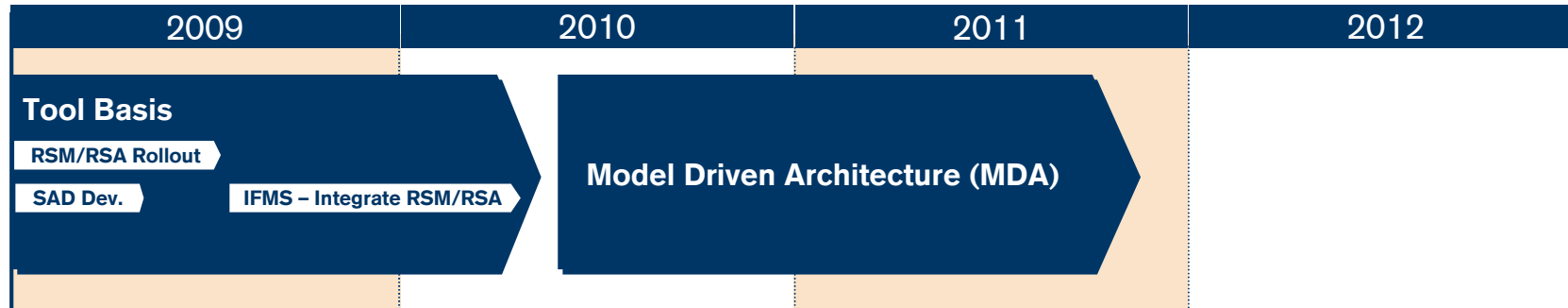
4

**Questions**

### 3. CS Solution Engineering Strategy Roadmap: Component Reuse & Software Product Lines



### 3. CS Solution Engineering Strategy Roadmap: Model Driven Architecture (MDA)



**Provide necessary infrastructure for MDA**

- Professional setup of RSM/RSA
- UML standards for design and architecture documentation

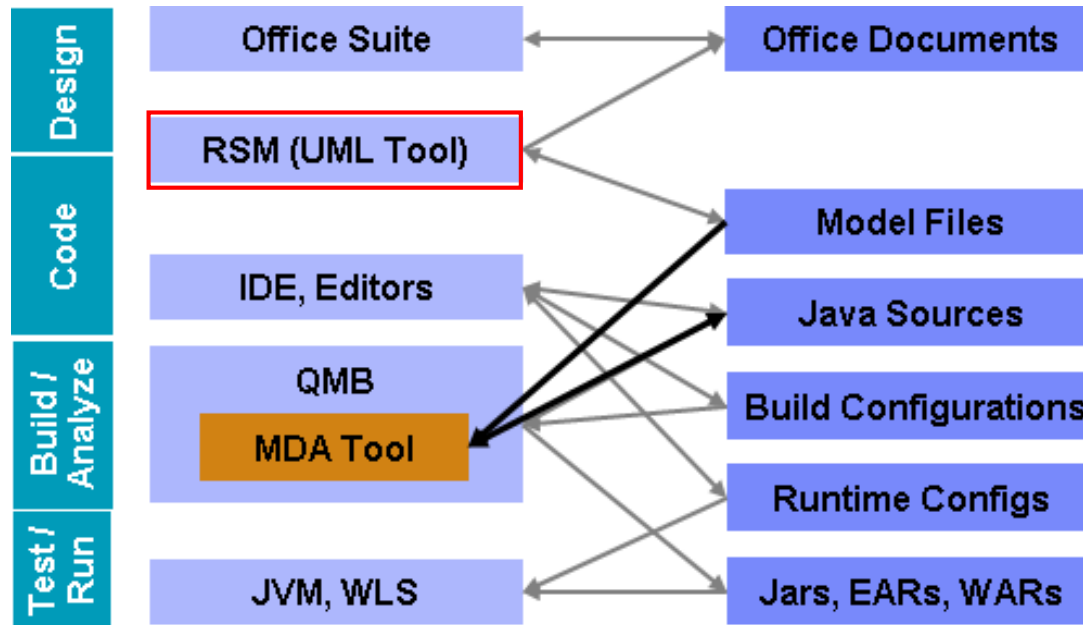


**Standardized and broadly applied generated SWE**

- Standard Code Generator (enable MDA out of RSM/RSA)
- Enhance standardized models and establish MDA-readiness

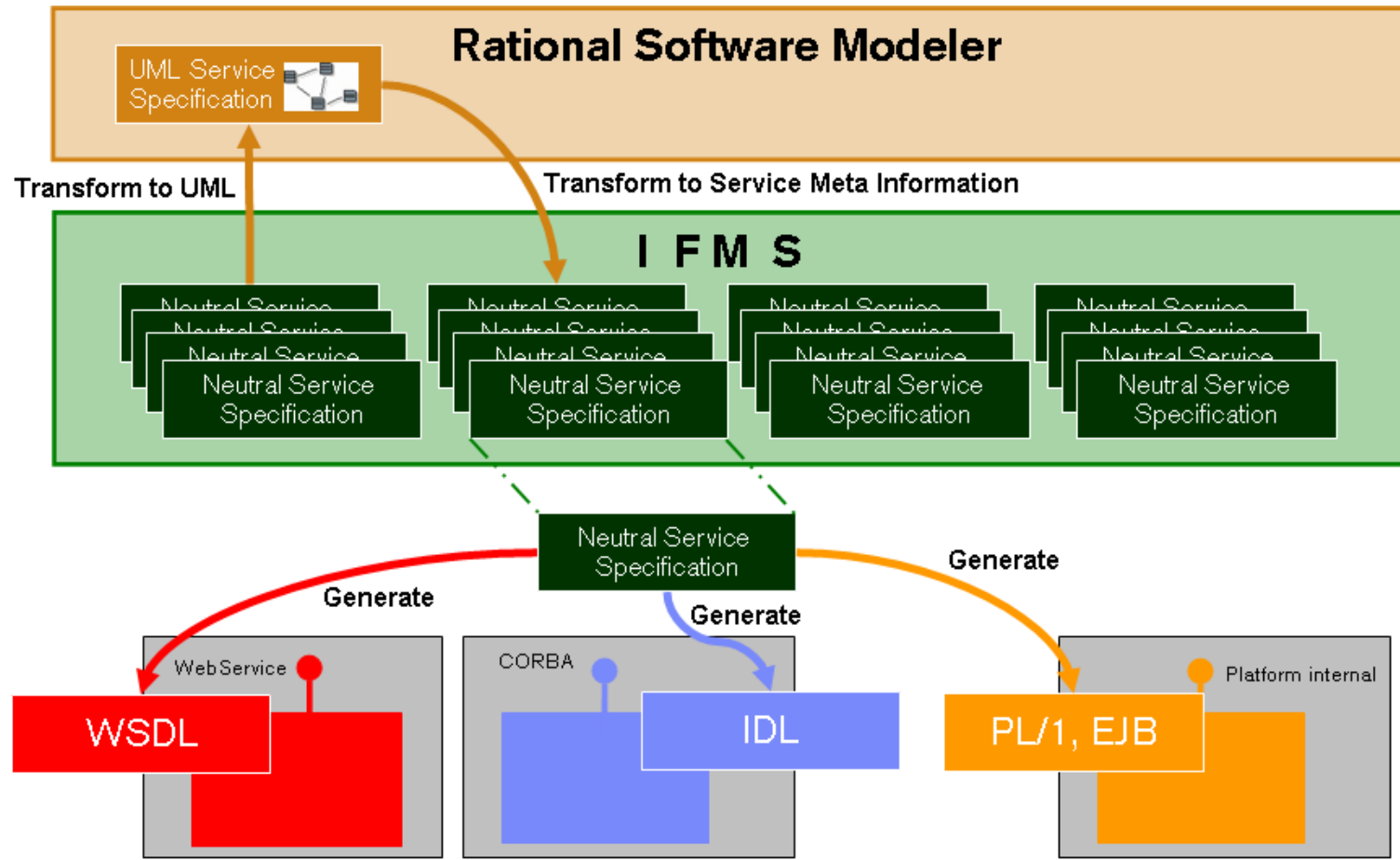
### 3. Long-term Concept for Model Driven Architecture at Credit Suisse IT for Java

- Long-term MDA concept (full integration of MDA tool needed)



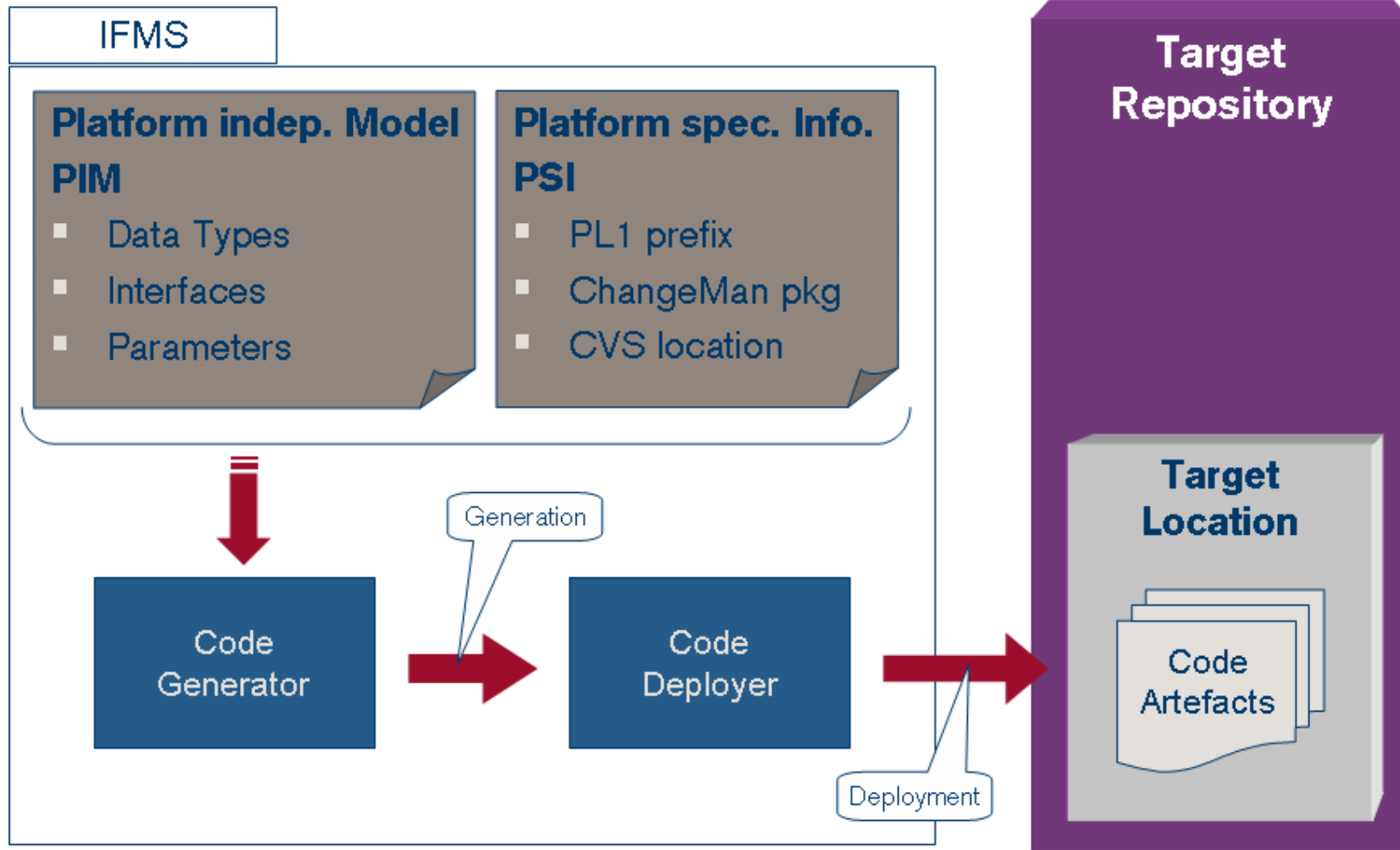
- Models are sources and input to the build process
- MDA-based generation is conducted on the central build server
- MDA tool(s) are part of the QMB build tool and are integrated "headless" (without direct user interface and interaction)

### 3. Model Driven Architecture (MDA) Concept for Services with the Interface Management System (IFMS) at Credit Suisse

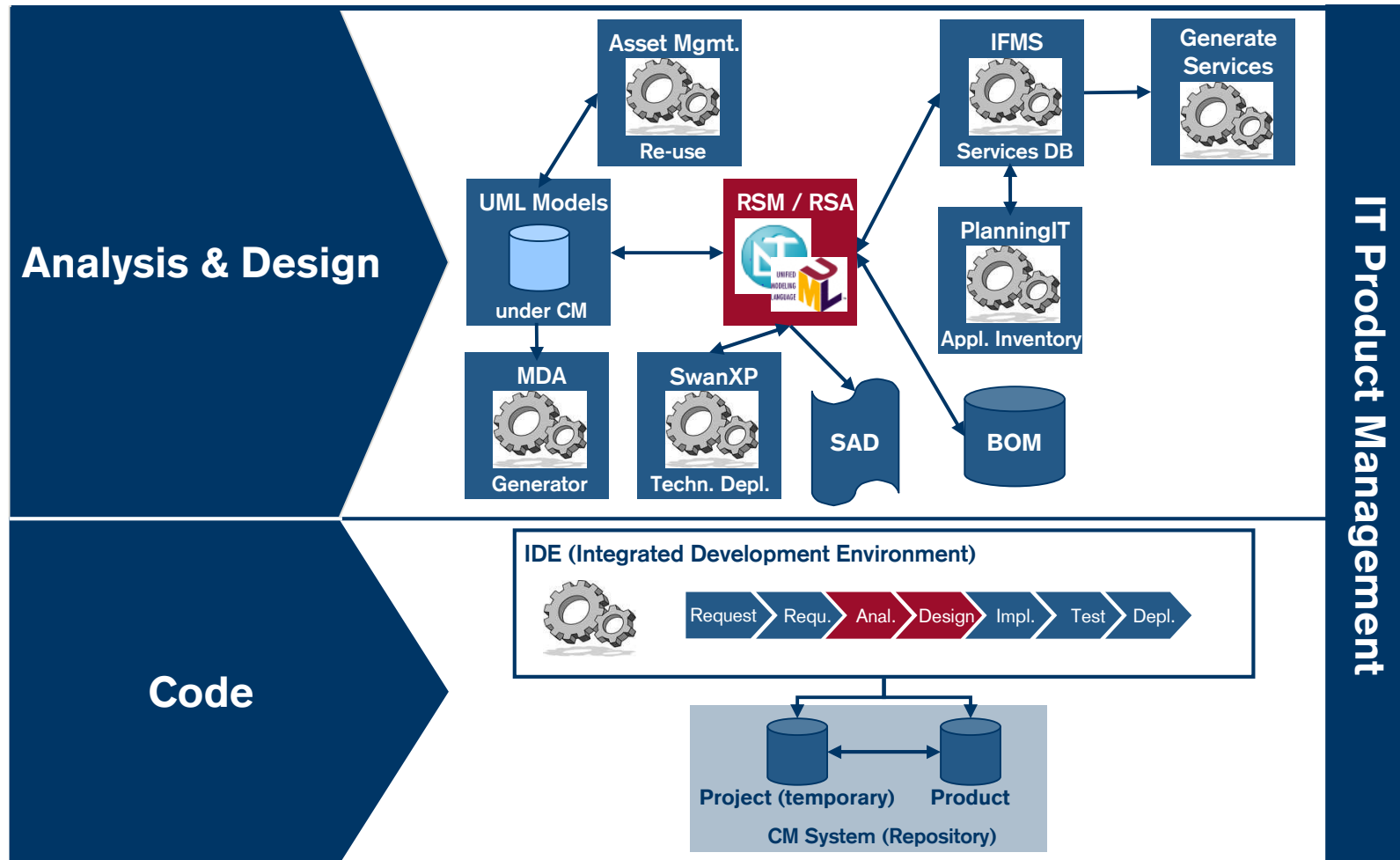




### 3. Model Driven Architecture for Services with IFMS: Generator Overview



### 3. Overview of future Rational Software Modeler Integration in the Context of the CS IT Tool-Landscape



# Agenda

1

**A brief Introduction** to Credit Suisse IT, the Credit Suisse IT Solution Delivery Standard Process and the Credit Suisse IT Standard Tool-Chain

2

**The Software Architecture Document (SAD)** – Next Generation of Software Modeling at Credit Suisse with Rational Software Modeler

3

**An Outlook to the Future:** Rational Software Modeler at Credit Suisse IT

4

**Questions**

## 4. Questions



# Appendix

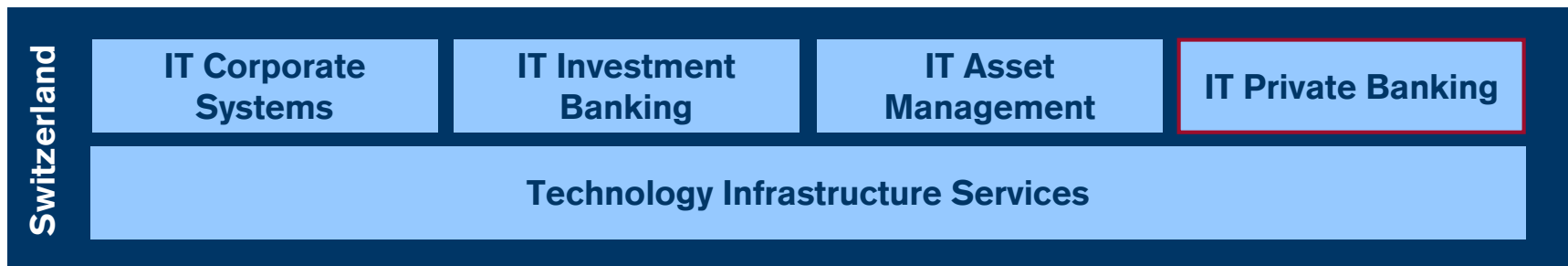
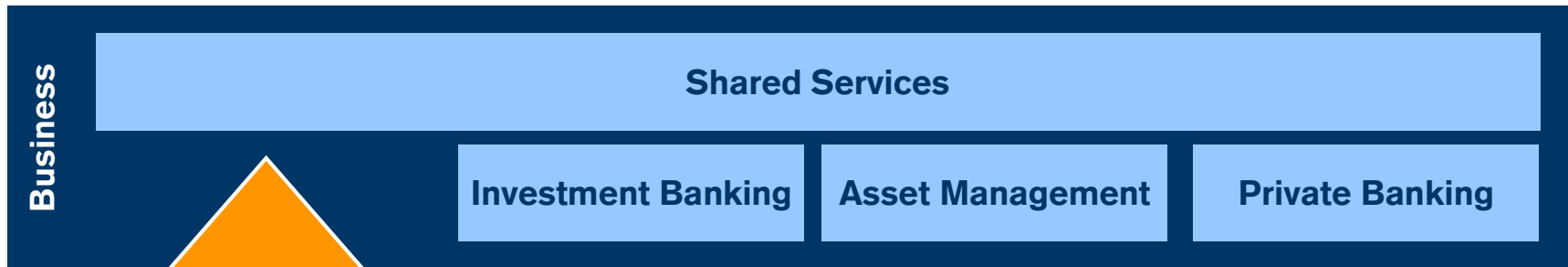
# Abbreviations

<b>aBOM</b>	<b>Application Business Object Model</b>
<b>Appl.</b>	<b>Application</b>
<b>BOM</b>	<b>Business Objects Model</b>
<b>CH</b>	<b>Switzerland</b>
<b>CM</b>	<b>Configuration Management</b>
<b>CS</b>	<b>Credit Suisse</b>
<b>IDE</b>	<b>Integrated Development Environment</b>
<b>IDL</b>	<b>Interface Definition Language</b>
<b>IFMS</b>	<b>Interface Management System</b>
<b>JVM</b>	<b>Java Virtual Machine</b>
<b>MDA</b>	<b>Model Driven Architecture</b>
<b>ML3</b>	<b>Maturity Level 3</b>
<b>PB</b>	<b>Private Banking</b>
<b>PIM</b>	<b>Platform Independent Model</b>
<b>RSA</b>	<b>Rational Software Architect</b>
<b>RSM</b>	<b>Rational Software Modeler</b>
<b>RUP</b>	<b>Rational Unified Process</b>
<b>QMB</b>	<b>QMBridge</b>

<b>SAD</b>	<b>Software Architecture Document</b>
<b>UI</b>	<b>User Interface</b>
<b>UML</b>	<b>Unified Modeling Language</b>
<b>WLS</b>	<b>WebLogic Server</b>
<b>WSDL</b>	<b>Web Service Definition Language</b>

# Definitions

<b>QMBridge (QMB)</b>	<p>A tool used for creating, building, testing, and releasing Java J2EE applications on the Credit Suisse standard application platform JAP. [Credit Suisse IT]</p>
<b>Model Driven Architecture (MDA)</b>	<ul style="list-style-type: none"> <li>▪ Standard defined by <b>OMG (Object Management Group)</b></li> <li>▪ Provides an open, vendor-neutral approach to the challenge of business and technology change</li> <li>▪ Separates business and application logic from underlying platform technology</li> <li>▪ Platform independent-models document the behavior of an application separated from the technology-specific code that implements it</li> <li>▪ Platform-independent models of an application can be realized on different platforms</li> <li>▪ Business and technical aspects of an application can each evolve at its own pace</li> </ul> <p>[OMG]</p>
<b>Business Object Model (BOM)</b>	<p>A <b>Business Object Model</b> is a conceptual model and is used for the communication between business and IT. The model captures all information need of the business by considering structural aspects and behavioral aspects. The model is expressed in the <b>UML-Notation</b>, e.g. class diagrams for the static structural modeling; state diagrams, sequence diagrams, activity diagrams etc. for the behavioral modeling. [Credit Suisse]</p>
<b>Interface Management System (IFMS)</b>	<p>Credit Suisse in-house solution for managing, cataloguing and generating services such as <b>WebServices, Corba-Services</b>, etc. [Credit Suisse]</p>



EMEA

APAC

Americas



