



# IBM SOA Summit



\* Informations valorisées et SOA,  
le couple gagnant.



IBM SOA\*  
Summit



**SOA Governance  
Rational Method Composer**

**Francis Geysermans**  
SOA Architect



# What is governance?



Establishing **chains of responsibility, authority and communication** to empower **people (decision rights)**  
Establishing **measurement, policy and control mechanisms** to enable **people to carry out their roles and responsibilities**





# What is Governance?

It's all part of Corporate Governance

## What is IT governance?

Establishing decision making rights associated with IT

Establishing mechanisms and policies used to measure and control the way IT decisions are made and carried out

## What is SOA governance?

Extension of IT governance focused on the lifecycle of services to ensure the business value of SOA



***SOA Governance is a catalyst for improving overall IT governance***



# Rational Method Composer

## ■ What is Rational Method Composer?

- A next generation process solution with *tooling*, *process libraries*, and *unified method architecture*
- It's a tool that enables you to create and publish a method that is usable and useful on your projects.

## ■ How does Rational Method Composer help you to do this?

- It supports a broad set of project types (small, medium, large, packaged application development, maintenance).
- It addresses enterprise needs (business, process, and systems engineering content).
- It allows you to reuse or customize pieces of existing content, as well as to add new content.
- It allows you to use your content to organize and publish a method Web site.
- It bridges the gap between process and project management by integrating with Rational Portfolio Manager (RPM).



The IBM Rational Method Composer 7.1 package contains

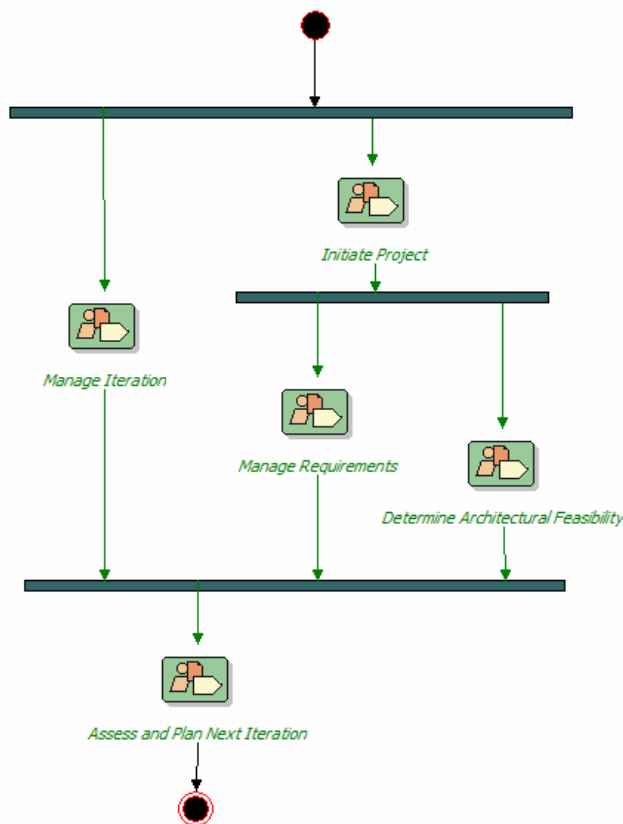
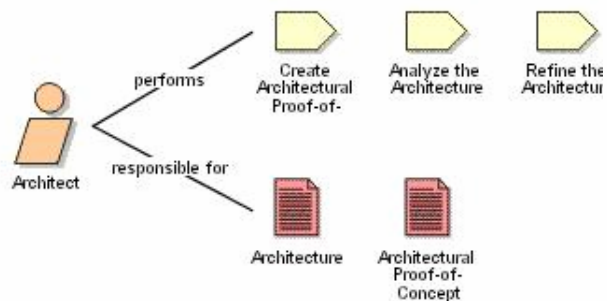
- **IBM Rational Method Composer 7.1 tool** (*the focus of this section*)
- **Method Library with the following plug-ins**
  - Rational Unified Process (*used for examples, exploration and labs in this course*)
  - Base Concepts
  - RUP Formal Resources
  - RUP Informal Resources
  - Business Modeling
  - Service-Oriented Architecture
  - RUP for J2EE
  - Rational Software Architect
  - Legacy Evolution
  - Rational Application Development
- **Published RUP Configurations**
  - RUP for Large Projects
  - RUP for Small Projects



## Role, Tasks, and Artifacts

## Activity Diagram

## Work Breakdown Structures



Presentation Name	Index
Basic Unified Process	0
Inception Phase - Iteration n	1
Manage Iteration	2
Initiate Iteration	3
Conduct daily meeting	4
Initiate Project	5
Define Vision	6
Plan the Project	7
Manage Requirements	8
Describe Requiremen	9
Find and Outline Acto	10
Detail Use Cases	11
Determine Architectural F	12
Analyze the Architec	13
Create Architectural	14
Assess and Plan Next Ite	15
Assess Results	16
Prioritize Work	17
Plan Next Iteration	18
Refine Project Plan	19

# Terminology: Method, Method Content and



A **Method** provides both the descriptions of work and the order of work. A method is end-to-end and is usable on a project. An example of a method is RUP.

**method = method content + process**

**Method content** provides descriptions of work that can be reused as important building blocks. These are the descriptions of tasks, roles, work products, guidelines and so on that are involved in completing work.

**Processes** provide the order of doing work. They do so by providing the order for the method content.

*Key difference: RUP toolset did not separate method content and process; different processes were difficult to represent; method content was difficult to reuse*





# Use RMC to Create Useful and Usable



The tool contains:

- Content A
- Content B
- Content A
- Content C
- Content A
- Content D
- Content K
- Content E
- Content I
- Content G
- Content M
- Content H
- Content N
- Content I
- Content O
- Content J
- Content P
- Content Q
- Content R
- Content S
- Content T
- Content U

- Process 1
- Process 2
- Process 3
- Process 4
- Process 5
- Process 6
- Process 7
- Process 8

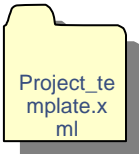
You make selections, changes and additions to specify your method

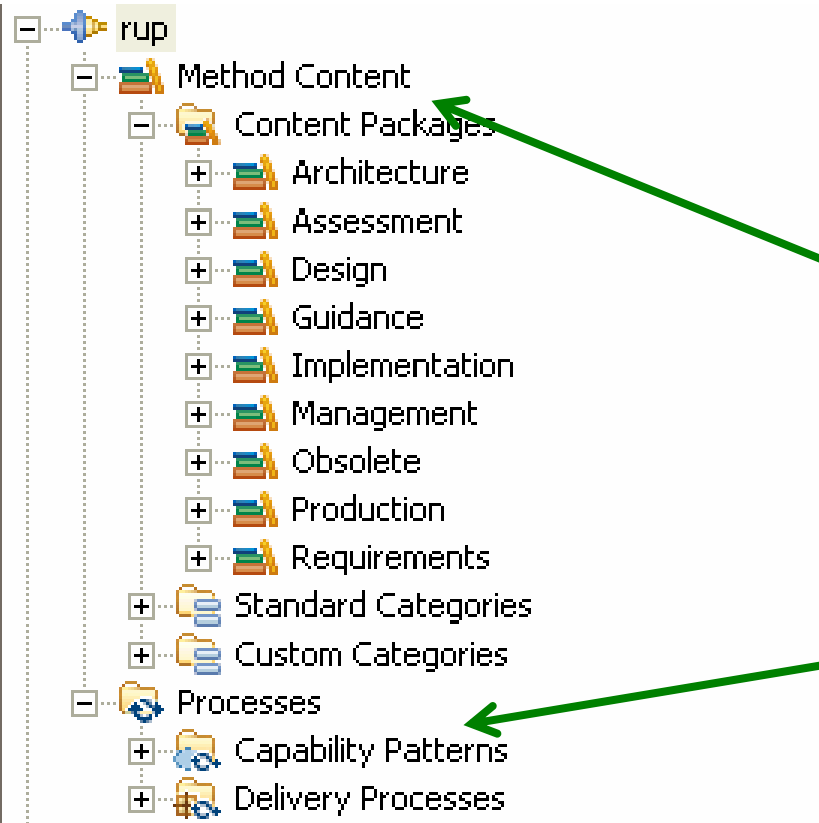
You can export a project template that is importable into IBM Rational Portfolio Manager

Your method:

- Your Content A'
- Your Content C'
- Your Content D'
- Your Content F'
- Your Content J'
- Your Content N'
- Your Content M'
- Your delivery process

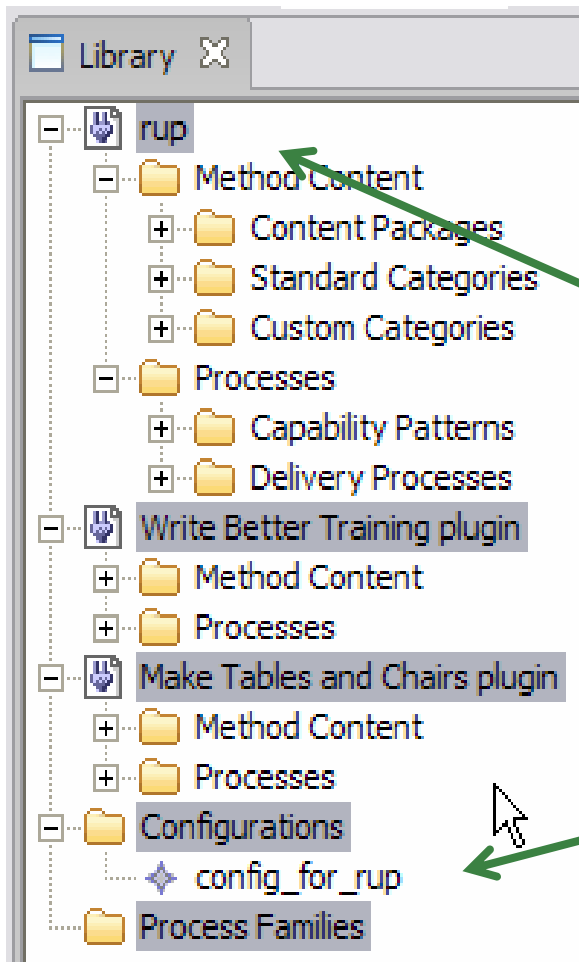
You now have a published method Web site that contains your content and that is organized by your delivery process





- Can be thought of as a “piece” of a method.
- Every plug-in has the same organizational structure:
- **Method Content:**
  - Content packages (where all method content elements are defined)
  - Standard Categories
  - Custom Categories
- **Processes**
  - Capability Patterns
  - Delivery Processes





- **Libraries contain**

- Method plug-ins
- Configurations

- **This sample library has:**

- Three method plug-ins
  - RUP
  - Write Better Training
  - Make Tables and Chairs

- **One configuration**

- config\_for\_RUP

# Terminology: Method Configuration



The screenshot shows two windows from IBM SOA Studio. The left window, titled 'Library', displays a tree view of method elements. The right window, titled 'Capability Pattern: construction-phase', shows the configuration for a specific method.

**Library Window:**

- rup
  - Method Content
    - Content Packages
    - Standard Categories
    - Custom Categories
  - Processes
    - Capability Patterns
    - Delivery Processes
- formal\_resources
- informal\_resources
- rup\_ux\_modeling\_plugin
- rup\_rad\_plugin
- rup\_rsa\_plugin
- base\_concepts
- irup\_accessibility\_plugin
- irup\_globalization\_plugin
- My plugin
- Configurations
  - config\_for\_rup (highlighted with a red circle)
  - config\_for\_formal
  - config\_for\_rup\_legacy\_evol\_plugin

**Configuration Window:**

Configuration: config\_for\_rup

Configuration Content

Please select method elements for this configuration.

Configuration Components:

- rup
  - Method Content
    - Content Packages
  - Processes
    - Capability Patterns
    - Delivery Processes
- formal\_resources
- informal\_resources
- rup\_ux\_modeling\_plugin
- rup\_rad\_plugin
- rup\_rsa\_plugin
- base\_concepts
- irup\_accessibility\_plugin
- irup\_globalization\_plugin

Green arrows point from the 'Content Packages', 'Processes', 'Capability Patterns', and 'Delivery Processes' folders in the Library to their respective checked items in the Configuration Components list. A red circle highlights 'config\_for\_rup' in the Library, with a red callout box pointing to it.

A configuration allows you to select or deselect from the content packages and processes available in your library's set of plug-ins.

Your selections help to determine the content of your published method Web site.

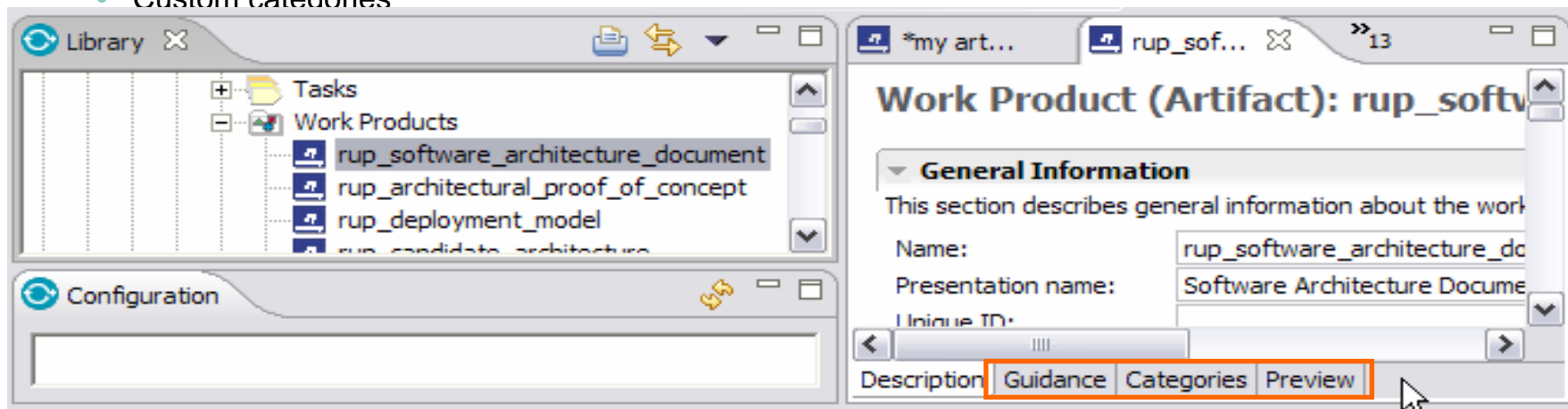
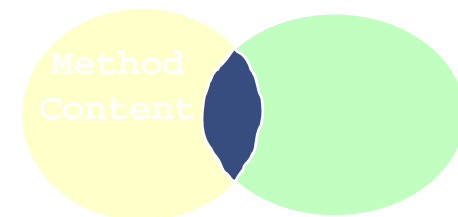
A configuration is given a name and is saved. It can be changed in the future, and re-published.

Named and saved configuration

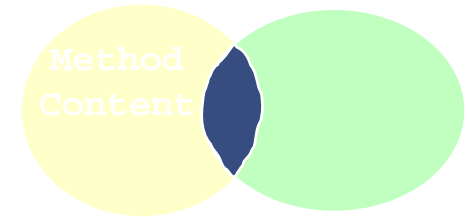


# Terminology: Work Product

- Anything used, produced, or modified by a task
- Types: deliverable, artifact, outcome
- A work product:
  - May reference guidance
  - May be categorized (categories discussed later):
    - Domains standard category
    - Work Product Types standard category
    - Custom categories

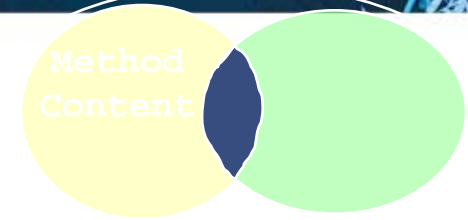


- **Defines a set of related skills, competencies, and responsibilities**
- **A role:**
  - May reference guidance
  - May be categorized (categories discussed later):
    - Role Sets standard category

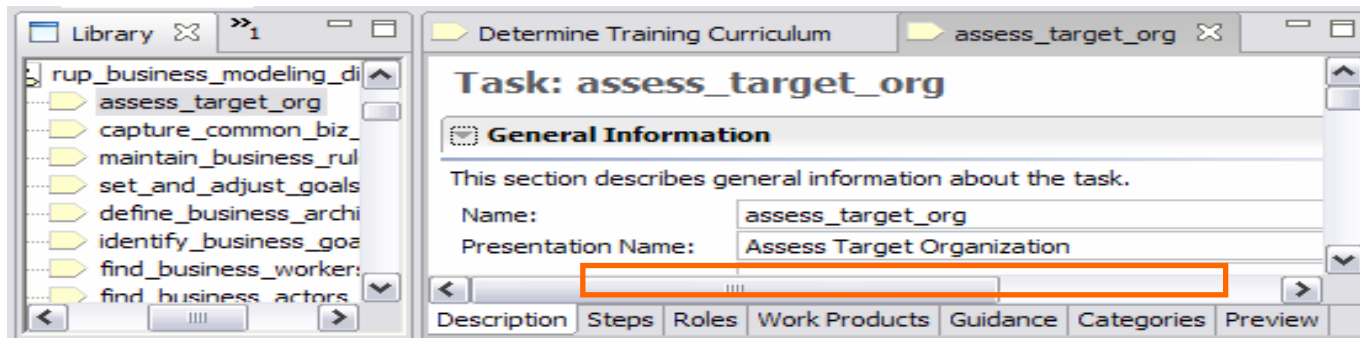


- **A role relates to work products as:**
  - Is Responsible For
  - Modifies

The screenshot displays a software interface with two main windows. The left window, titled 'Library', shows a hierarchical tree structure with folders for 'Content Packages', 'Architecture', 'Roles', 'Tasks', and 'Work Products'. Under 'Roles', the role 'rup\_software\_architect' is selected. The right window, titled 'rup\_software\_architect', shows the configuration details for this role. It includes a 'General Information' section with fields for 'Name' (rup\_software\_architect), 'Presentation name' (Software Architect), and 'Brief description' (This role leads the development of the sy... includes promoting and creating support t... constrain the overall design and impleme...). At the bottom of the right window, there are tabs for 'Description', 'Work Products', 'Guidance', 'Categories', and 'Preview'. The 'Work Products' tab is currently selected and highlighted with an orange border.



- **Describes work being performed by roles**
- **A task:**
  - Has steps
  - May reference guidance
  - May be categorized (categories discussed later):
    - Discipline standard category
    - Custom categories
- **A task uses work products as its:**
  - Mandatory input
  - Optional input
  - Output
- **A task is performed by roles:**
  - Primary performer
  - Additional performer



# Terminology: Method Elements and their Relationships

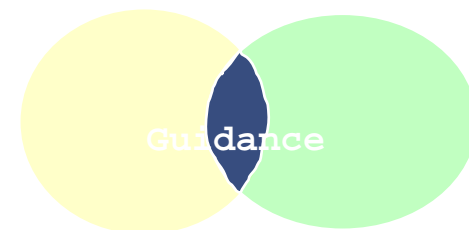


	Task	Work Product	Role
<p>Task</p> <ul style="list-style-type: none"> <li>•May refer to Guidance</li> <li>•Has Steps</li> <li>•Is in Categories                             <ul style="list-style-type: none"> <li>•Disciplines standard category</li> <li>•Custom Categories</li> </ul> </li> </ul>	<ul style="list-style-type: none"> <li>•Variability (contributes, replace, extends)</li> </ul>	<ul style="list-style-type: none"> <li>•Mandatory Inputs</li> <li>•Optional Inputs</li> <li>•Outputs</li> </ul>	<ul style="list-style-type: none"> <li>•Primary Performer</li> <li>•Additional Performer</li> </ul>
<p>Work Product</p> <ul style="list-style-type: none"> <li>•May refer to Guidance</li> <li>•Is in Categories:                             <ul style="list-style-type: none"> <li>•Domains standard category</li> <li>•Work Product Types standard category</li> <li>•Custom Categories</li> </ul> </li> </ul>	<ul style="list-style-type: none"> <li>•Mandatory Input</li> <li>•Optional Input</li> <li>•Output</li> </ul>	<ul style="list-style-type: none"> <li>•Artifacts can contain artifacts</li> <li>•Variability (contributes, replace, extends)</li> </ul>	<ul style="list-style-type: none"> <li>•Responsible Role</li> <li>•Modifying Roles</li> </ul>
<p>Role</p> <ul style="list-style-type: none"> <li>•May refer to Guidance</li> <li>•Is in Categories:                             <ul style="list-style-type: none"> <li>•Role Sets standard category</li> <li>•Custom Categories</li> </ul> </li> </ul>	<ul style="list-style-type: none"> <li>•Primary Performer</li> <li>•Additional Performer</li> </ul>	<ul style="list-style-type: none"> <li>•Role is Responsible For</li> <li>•Role Modifies</li> </ul>	<ul style="list-style-type: none"> <li>•Variability (contributes, replace, extends)</li> </ul>





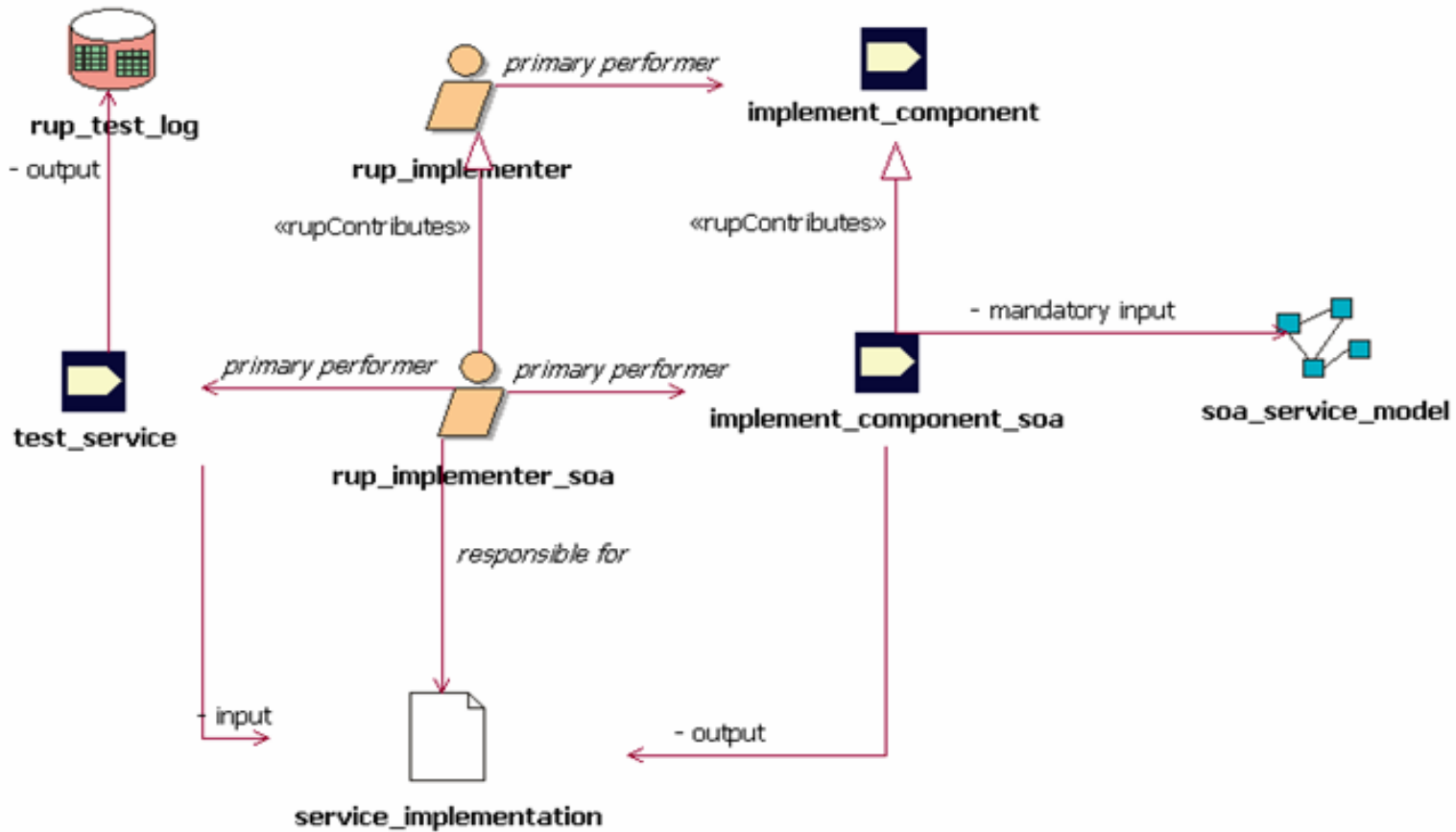
- **Guidance can be attached to both method and process elements in order to provide additional guidance about those elements.**
- **Types**
  - Checklist
  - Concept
  - Example
  - Guideline
  - Practice
  - Report
  - Reusable Asset
  - Roadmap
  - Supporting Material
  - Template
  - Term Definition
  - Tool Mentor
  - White Paper



- **Content variability refers to a mechanism in the tool that allows you to customize method content without directly modifying the original content.**
- **Content variability allows you to change method content in plug-in A by building plug-in B that **contributes**, **extends**, or **replaces** elements. This keeps the original elements in plug-in A intact since all the changes are defined in your plug-in B.**
- **Content variability is useful, for example:**
  - To change the description of an existing role
  - To add steps to an existing task

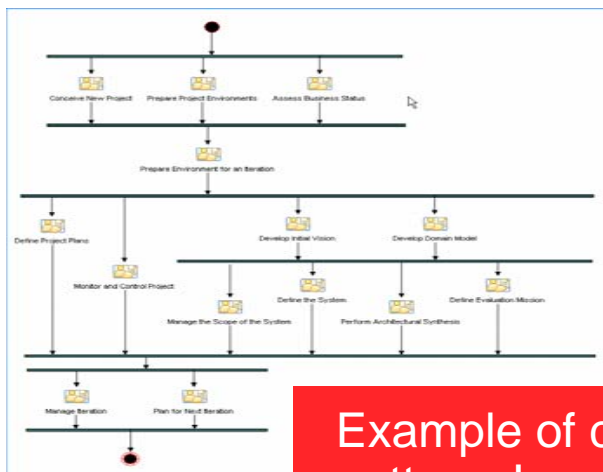
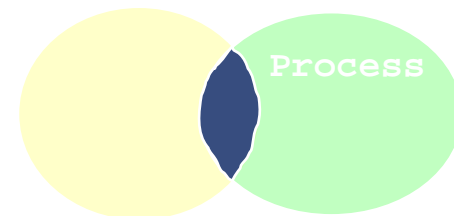


# Using Variability in a New Configuration

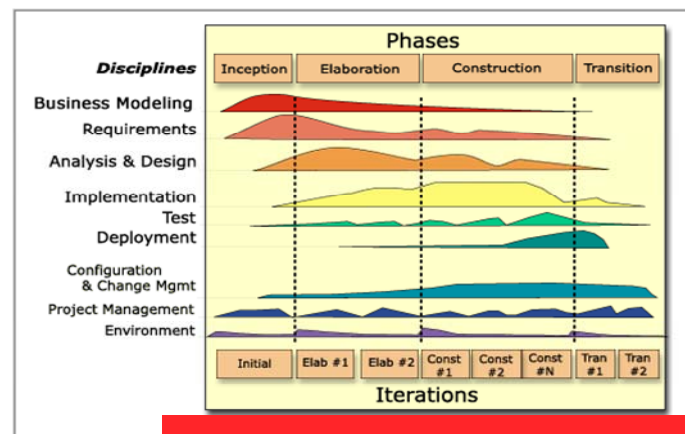


# Terminology: Process

- Defines how method content can be assembled into an executable process with a defined lifecycle
- Specific to the scale/context of project (e.g. greenfield vs. maintenance, formal/high ceremony vs. agile)
- Consists of two types:
  - End-to-end complete project lifecycles (**Delivery Processes**)
  - Process fragments that can be used to compose Delivery Processes (**Capability Patterns**)



Example of capability pattern: Inception iteration workflow

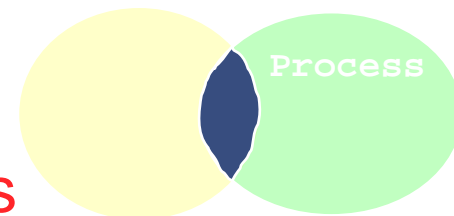


Example of delivery process: RUP lifecycle



# Terminology: Process Views

- **Process views are hierarchical arrangements of content elements that define the process**
- **Structured using work units called activities**
  - The lowest levels of process views are references to core method elements
- **There are four process views:**
  - **Work Breakdown Structure (WBS)** (shows tasks needed to perform the activities of the process)
  - **Team Allocation (TA)** (shows roles that perform the tasks in the WBS)
  - **Work Product Usage (WPU)** (shows work products for the tasks in the WBS)
  - **Consolidated View** (shows tasks, roles and work products)



- **Rational Method Composer is a configurable process platform that helps solve development challenges facing:**
  - Executives
    - Provide consistent results across all projects through proven best practices integrated with practitioner tools
  - Project Managers
    - Adapt the process to the unique needs of your project
  - Practitioners
    - Get the guidance to get the job done.
    - Personalize the process to address the specific development challenges, technologies and tools you face.

Thank  
YOU

