Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte
Campus Natal-Central
Diretoria Acadêmica de Gestão e Tecnologia da Informação
Curso de Tecnologia em Análise e Desenvolvimento de Sistemas

Eclipse Process Framework

Disciplina: Processo de Desenvolvimento de Software

Prof. Fellipe Aleixo (*fellipe.aleixo@ifrn.edu.br*)

Visão Geral do Projeto EPF

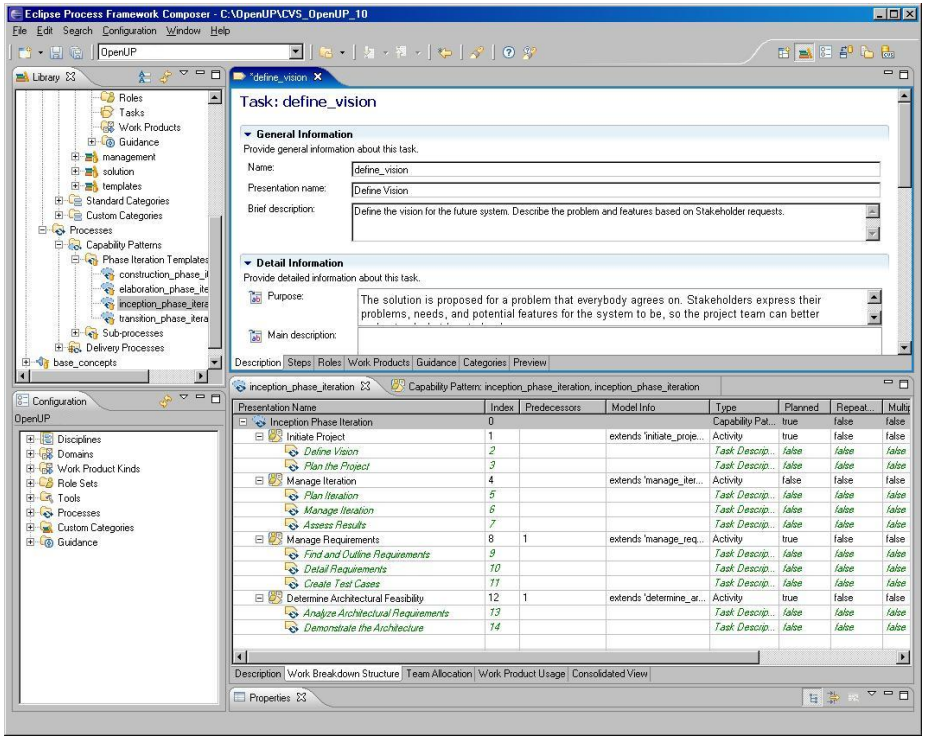
- É um projeto open source da Fundação Eclipse
- Iniciou em janeiro de 2006
- Tem como objetivo prover:
  - Um framework e ferramentas para edição, configuração e publicação de processos de software
  - Processos modelo
- EPF não é:
  - Aplicável apenas ao desenvolvimento Java no Eclipse
  - Voltado a criação do “processo perfeito”

Objetivos do EPF

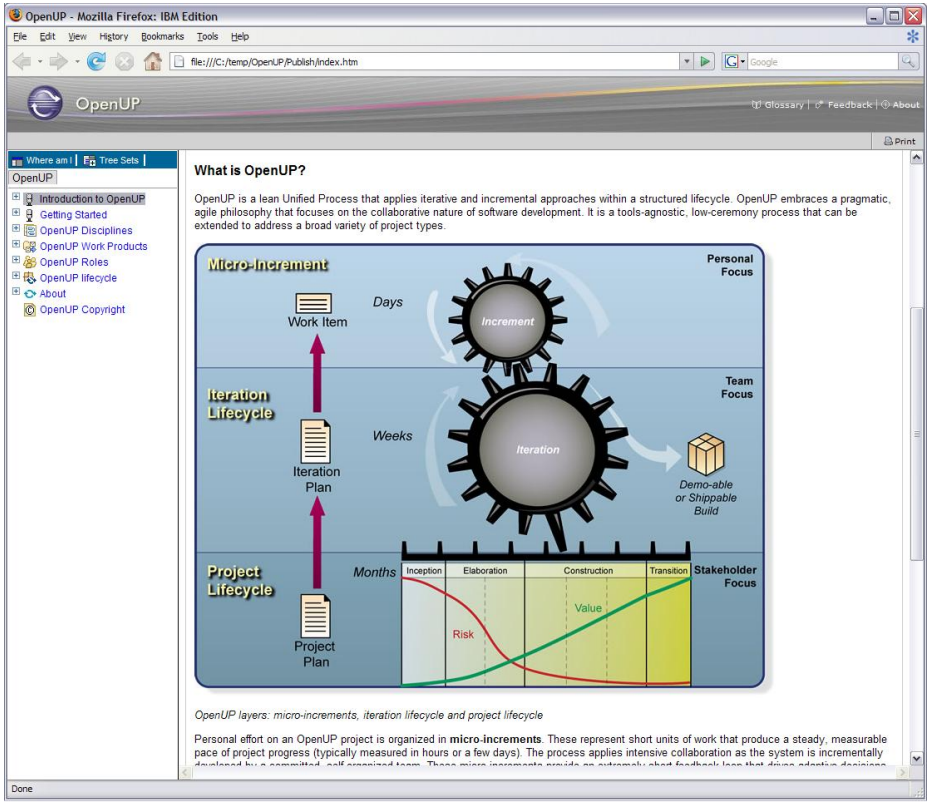
- Prover um framework customizável para engenharia de processos de software
  - Prover um framework extensível e um conjunto de ferramentas "modelo" para engenharia de processos
  - Prover um conteúdo de processo par um conjunto de processos de desenvolvimento e gerenciamento de software
    - suportando iterativo, ágil e incremental
    - que possa ser extensível

Objetivos do EPF

Ferramenta Modelo:
EPF Composer



Exemplo de Processo Modelo:
OpenUP



A Abordagem EPF

Standardize representation and manage libraries of reusable **Method Content**

Content on agile development

Content on managing iterative development

Guidance on serialized java beans

JUnit user guidance

Content on J2EE

Configuration mgmt guidelines

Develop and manage **Processes** for performing projects

Process for Custom Application Development with J2EE

Process for Embedded System Development

Process for SOA Governance

Process assets patterns

Standard or reference processes

Enactable project plan templates

Corporate guidelines on compliance

Configure a cohesive process framework customized for my project needs

Create project plan templates for **Enactment** of process in the context of my project

Quem se Beneficia com o EPF?

1. Projetos individuais → usar os processos disponibilizados pelo EPF
2. Empresas de desenvolvimento → (i) adoção das "melhores práticas", (ii) incorporar as lições aprendidas, (iii) padronizar a linguagem dentro da organização e (iv) endereçar necessidades específicas
3. Academia → atuando nos papéis de "consumidor" e de responsável por trazer o estado-da-arte no desenvolvimento de software para a "industria"
4. Empresas de tecnologia → desenvolvimento de ferramentas adequadas aos processo EPF

Informações sobre o meta-modelo utilizado para armazenar às informações de um processo de software

REPRESENTAÇÃO DOS PROCESSOS NO EPF

Representação de Processos EPF

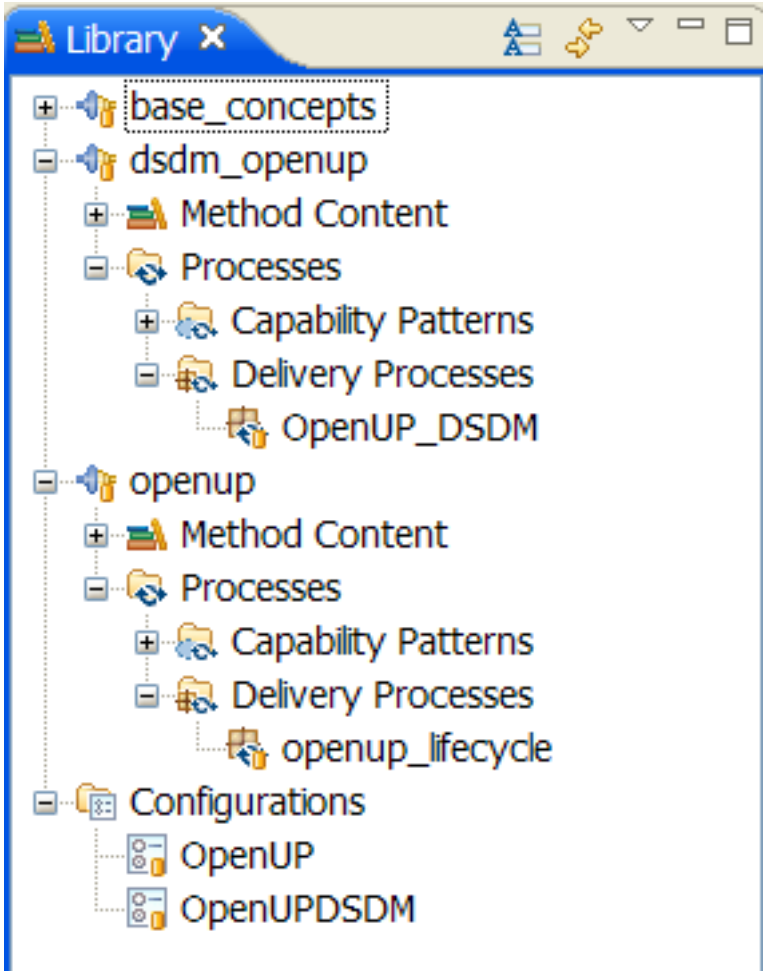
- O EPF armazena as informações de processo em um modelo segundo o meta-modelo UMA
  - UMA = *Unified Method Architecture*
  - O UMA foi desenvolvido com base no SPEM 1.0 e influenciou a criação do SPEM 2.0 (OMG)
  - SPEM = *Software Process Engineering Meta-model*
- Qualquer processo pode ser representado através destes meta-modelos

Conceitos Básicos: *Method Library*

- *Method Library*
  - Armazena um conjunto de elementos de método
- *Method Plug-in*
  - Representa um conjunto de "pacotes" de métodos e processos
- *Method Configuration*
  - Um subconjunto a lógico da *Method Library*
- *Delivery Process*
  - Uma abordagem completa e integrada para realização de tipo específico de projeto

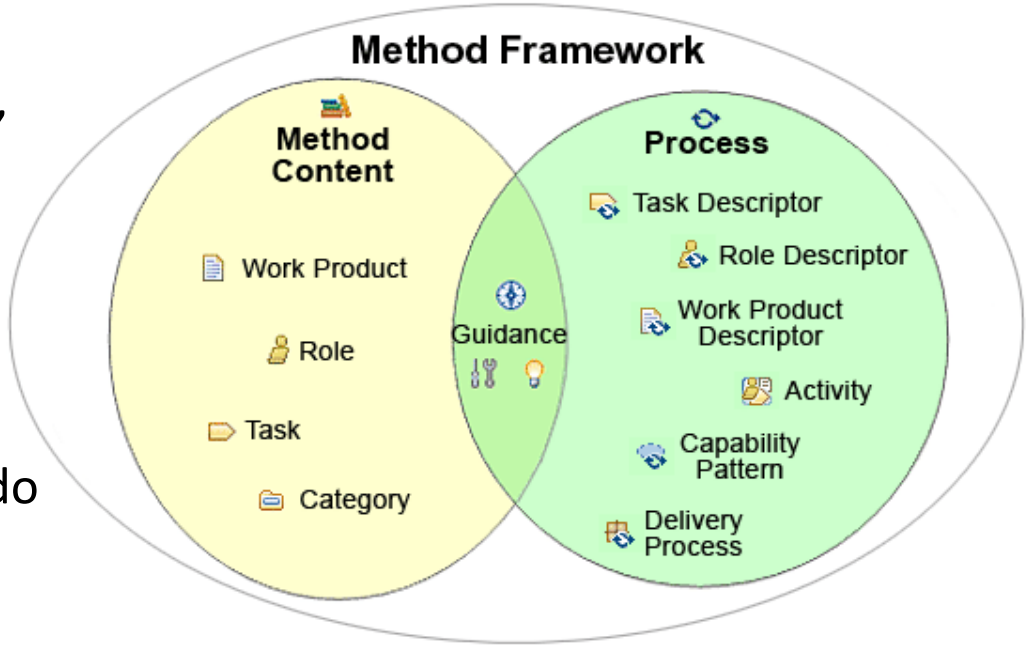
Conceitos Básicos: *Method Library*

- Bibliotecas de métodos contém:
  - *Method plug-ins*
  - Configurações
- A biblioteca do OpenUP possui:
  - Três *method plug-ins*
    - base_concepts
    - dsdm_openup
    - openup
  - Dois *delivery processes*
    - Openup_DSDM
    - openup_lifecycle
  - Duas configurações
    - OpenUP
    - OpenUPDSDM



Conceitos Básicos: *Method Content e Process*

- *Method Content* (quem, o que, por que, como)
  - Definição de papéis, tarefas, produtos de trabalho e seus relacionamentos
- *Process* (quando)
  - Sequência das Fases, Iterações, Atividades e *Milestones* → ciclo de vida do desenvolvimento
  - Define quando realizar as tarefas → diagrama de atividades e/ou *Work Breakdown Structures*

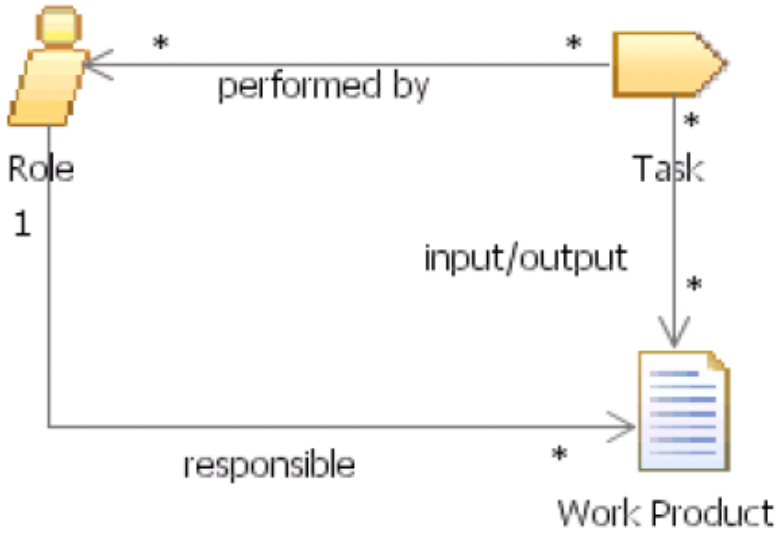


Method Content

CONTEUDO DE MÉTODO

(1) Papel

- Definem um conjunto de habilidades, competências e responsabilidades relacionadas
- Não representam indivíduos os quais podem desempenhar vários papéis
- Papéis realizam tarefas
- Papéis são responsáveis pode produtos de trabalho



(2) Produto de Trabalho

- Na maioria dos casos representam elementos tangíveis usados, modificados e produzidos por uma tarefa
- Papéis usam produtos de trabalho para realizar tarefas e produzir/atualizar outros produtos de trabalho
- São responsabilidade de um papel
- Podem ser de três tipos:
  -  Artefatos – item de configuração gerenciado
  -  Entregável – requisitado pelo *cliente/stakeholder*
  -  Resultado – efeito "intangível" de uma tarefa, como por exemplo um servidor ou ferramenta instalada

(3) Tarefa

- Define uma unidade de trabalho a ser atribuída (geralmente com a duração de algumas horas)
- Desempenhada por papéis (primário e opcionais)
- Possuem um propósito claro, e provêem uma descrição de passo-a-passo do trabalho necessário para completar a tarefa (atingir meta)
- Modifica ou produz produtos de trabalho
- Não definem quando as mesmas deverão ser executadas no ciclo de vida

(4) Orientações

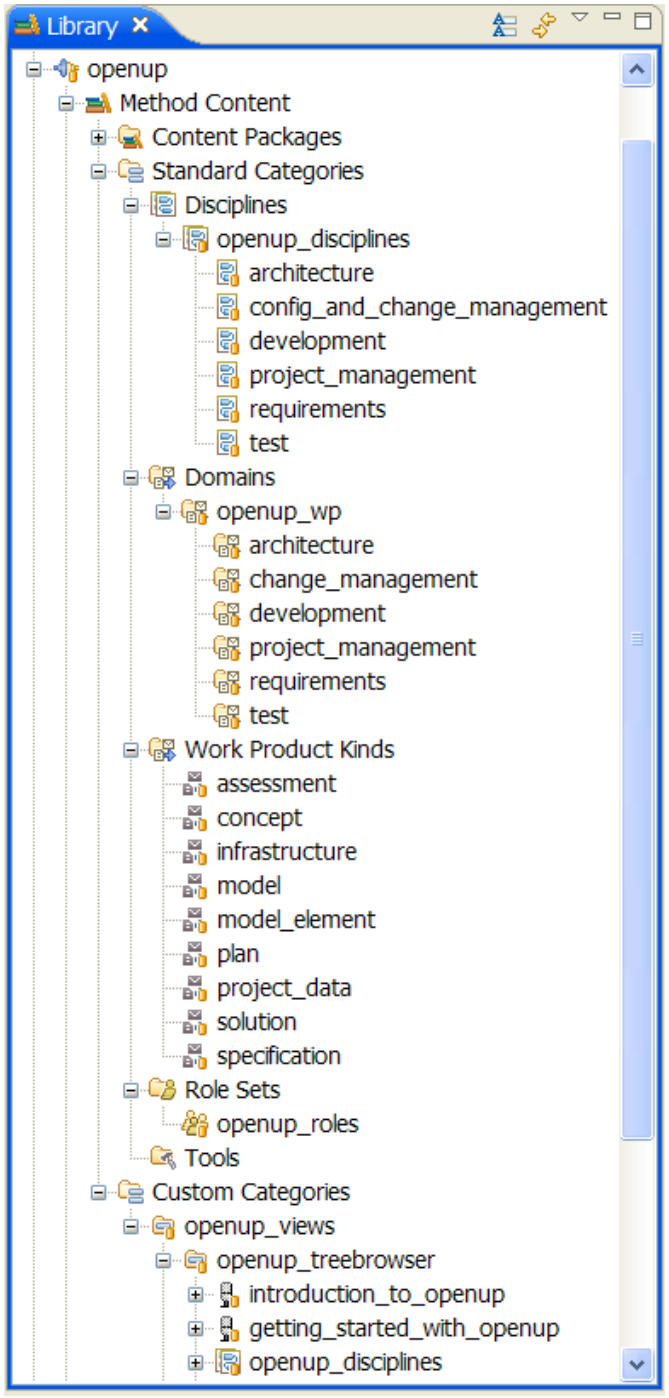
- Orientações podem ser associadas a um papel, tarefa e produtos de trabalhos
- Tipos diferentes para propósitos diferentes
- As orientações são usadas para detalhar a metodologia e oferecer informações de suporte
  - Tarefas dizem "o que" precisa ser feito, já as orientações detalham "como fazer"

Tipos de orientação:

- Checklist 
- Concept 
- Example 
- Guideline 
- Estimate 
- Considerations 
- Practice 
- Report 
- Reusable Asset 
- Roadmap 
- Supporting Material 
- Template 
- Term Definition 
- Tool Mentor 
- Whitepaper 

(5) Categorias

- Usadas para agrupar elementos de método relacionados
- São cinco as categorias padrão:
  - *Discipline*: agrupamento de tarefas
  - *Domain*: agrupamento de produtos de trabalho
  - *Work Product Kind*: similar ao *Domain*
  - *Role Set*: agrupamento de papéis
  - *Tool*: agrupamento de ferramentas
- Categorias pode ser aninhadas
- Podem ser definidas novas categorias
- Os elementos são categorizados através do editor de propriedades
- Usadas para criar visões no site web do processo



Conteúdo de processo

PROCESS CONTENT

(1) Padrão de Capacidade

- Definem a sequência de tarefas relacionadas, realizadas no intuito de alcançar um objetivo maior
- Tarefas pode ser especializadas para um dado contexto

Descritor de Tarefa

(instância de Tarefa)

Descritor de Papel

(instância de Papel)

Descritor de Produto de Trabalho

(instância de PT)

Capability Pattern: Initiate Project

This capability pattern bundles tasks required to define the vision and create a project plan.

Description | Work Breakdown Structure | Team Allocation | Work Product Usage

Expand All Sections | Collapse All Sections

Work Breakdown

Breakdown Element	Steps	Index	Predecessors	Model Info	Type	Planned	Repeatable	Multiple Occurrences	Ongoing	Event-Driven	Optional	Tea
Define Vision		1			Task Descriptor							
Analyst				Primary Performer	Role Descriptor	✔						
Stakeholder				Secondary Performer	Role Descriptor	✔						
Vision				Optional Input	Work Product Descriptor	✔						
Glossary				Output	Work Product Descriptor	✔						
Vision				Output	Work Product Descriptor	✔						
Plan the Project		2			Task Descriptor							
Project Manager				Primary Performer	Role Descriptor	✔						
Analyst				Secondary Performer	Role Descriptor	✔						
Architect				Secondary Performer	Role Descriptor	✔						
Stakeholder				Secondary Performer	Role Descriptor	✔						
Tester				Secondary Performer	Role Descriptor	✔						
Vision				Mandatory Input	Work Product Descriptor	✔						
Work Items List				Mandatory Input	Work Product Descriptor	✔						
Project Plan				Optional Input	Work Product Descriptor	✔						
Risk List				Optional Input	Work Product Descriptor	✔						
Project Plan				Output	Work Product Descriptor	✔						
Risk List				Output	Work Product Descriptor	✔						
Work Items List				Output	Work Product Descriptor	✔						

(1) Padrão de Capacidade

- Podem ser “aninhados” e visualizados de forma gráfica
- Uma atividade é uma instância de um padrão de capacidade

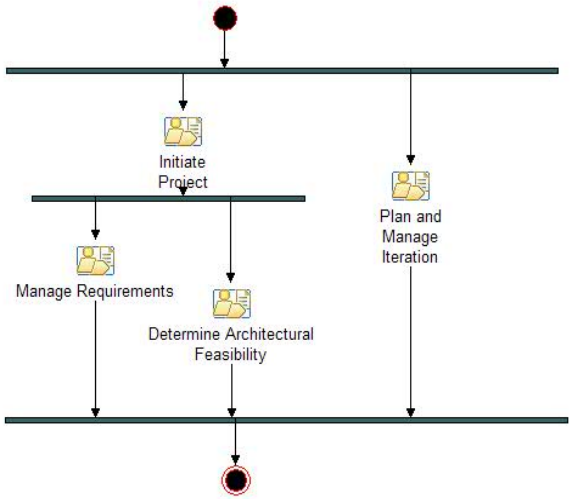
Capability Pattern: Inception Phase Iteration

This iteration template defines the activities (and associated roles and work products) performed in a typical iteration in the Inception phase. Each activity and related goals are described.

Description | Work Breakdown Structure | Team Allocation | Work Product Usage

Expand All Sections | Collapse All Sections

Workflow



Back to top

Work Breakdown

Breakdown Element	Steps	Index	Predecessors	Model Info	Type	Planned	Repeatable	Multiple Occurrences	Ongoing	Event-Driven	Optional	Team
Initiate Project		1			Activity	✔						
Plan and Manage Iteration		4			Activity				✔			
Manage Requirements		8	1		Activity	✔						
Determine Architectural Feasibility		12	1		Activity	✔					✔	

Back to top

Atividade

instância de padrão de capacidade

(2) Processo de Entrega

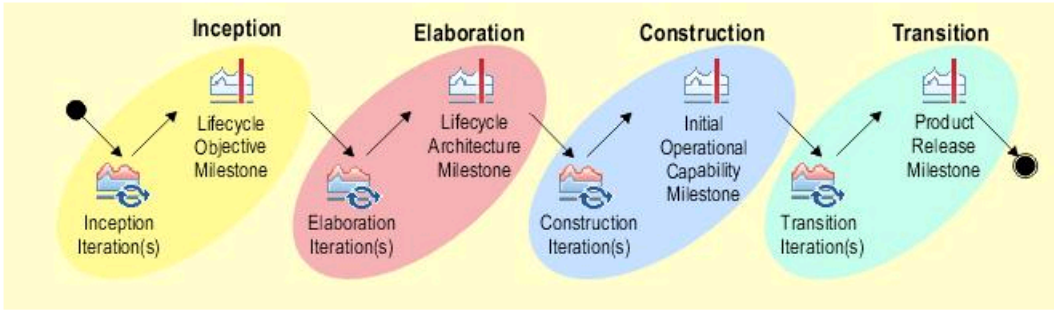
- Definidos usando uma Estrutura de Divisão de Trabalho (Work Breakdown Structure) e/ou diagramas de atividade
- Define o clico de vida completo para um processo
- Pode incluir iterações, Fases, Milestones (tipos de Atividades)

Delivery Process: OpenUP lifecycle

This delivery process defines an end-to-end software development lifecycle that supports the core principles of OpenUP. It is designed to support small, co-located teams in their daily activities.

Description | Work Breakdown Structure | Team Allocation | Work Product Usage

Workflow



Work Breakdown

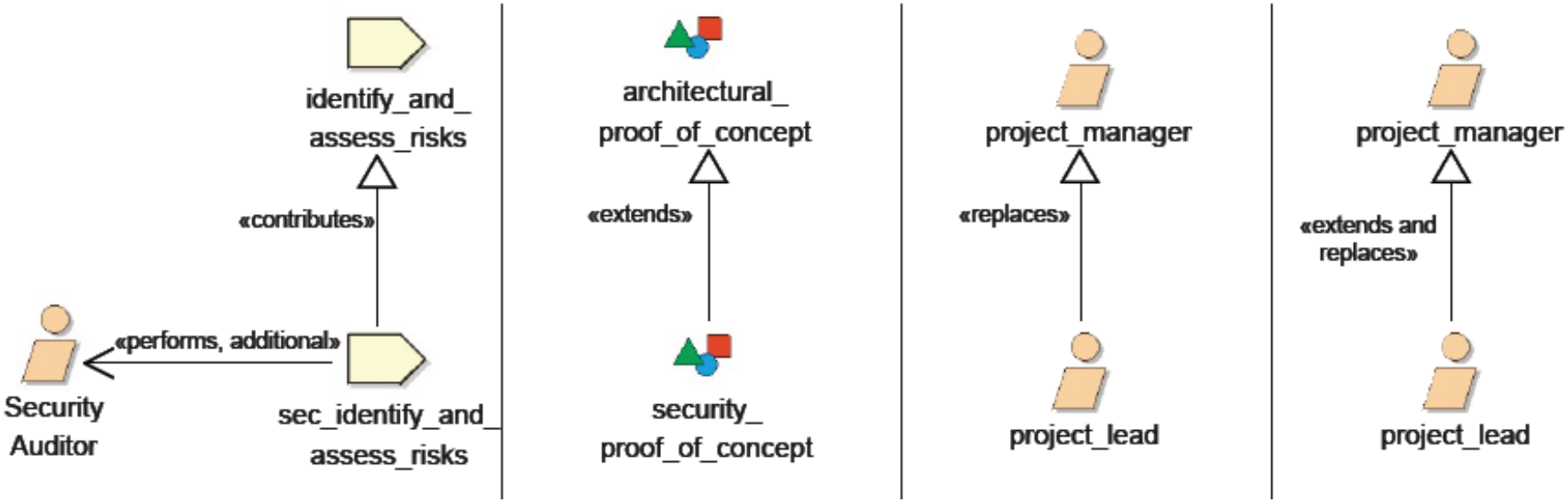
Breakdown Element	Steps	Index	Predecessors	Model Info	Type	Planned	Repeatable
Inception Iteration [1..n]		1			Activity	✔	
Initiate Project		2			Activity	✔	
Plan and Manage Iteration		5			Activity		
Identify and Refine Requirements		9	2		Activity	✔	
Agree on the Technical Approach		13	2		Activity	✔	
Lifecycle Objectives Milestone		15	1		Milestone	✔	
Elaboration Iteration [1..n]		16	15		Activity	✔	✔
Lifecycle Architecture Milestone		46	16		Milestone	✔	
Construction Iteration [1..n]		47	46		Activity	✔	✔
Initial Operational Capability Milestone		68	47		Milestone	✔	
Transition Iteration [1..n]		69	68		Activity	✔	✔
Product Release Milestone		86	69		Milestone	✔	

(3) Variabilidade de Método

- Mecanismo que permite a customização do conteúdo do método sem modificar o seu conteúdo original
- Similar ao conceito de herança (OO)
  - Permite o reuso através da especialização
- Por exemplo, se um *plugin* B estende um *plugin* A:
  - Os elementos originais do *plugin* A permanecem intactos – todas as mudanças são definidas pelo *plugin* B
- Variabilidade de conteúdo é útil, por exemplo, para:
  - Alterar a descrição de um papel já existente
  - Adicionar passos para uma tarefa já existente
  - Adicionar orientações a uma tarefa já existente

(3) Variabilidade de Método

- Quatro tipos de variabilidade de método:
  - *Contribute* → adicionam conteúdo aos elementos base
  - *Extends* → herdam o conteúdo do elemento base e especializam alguns ou todos (ambos são publicados)
  - *Replace* → substituem o elemento base (só o novo elemento é publicado)
  - *Extends-Replace* → similar ao *extends*, mas o elemento base não é publicado



(4) Variabilidade de Processo

- Mecanismo de reuso similar à variabilidade de método
- Adicionalmente, Atividades também pode ser criadas a partir dos padrões de capacidade, das seguintes formas:
  - *Extends* → a atividade herda as propriedades de um padrão de capacidade. Atualizações no padrão de capacidade são automaticamente refletidas para a atividade
  - *Copy* → a atividade é criada com base no padrão de capacidade. Mas, não permanecem sincronizados
  - *Deep Copy* → similar ao copy, mas aplicado de forma recursiva para as atividades
  - *Local Variability* → quando um padrão de capacidade é definido (por Extends ou Copy) variabilidade local pode ser realizada

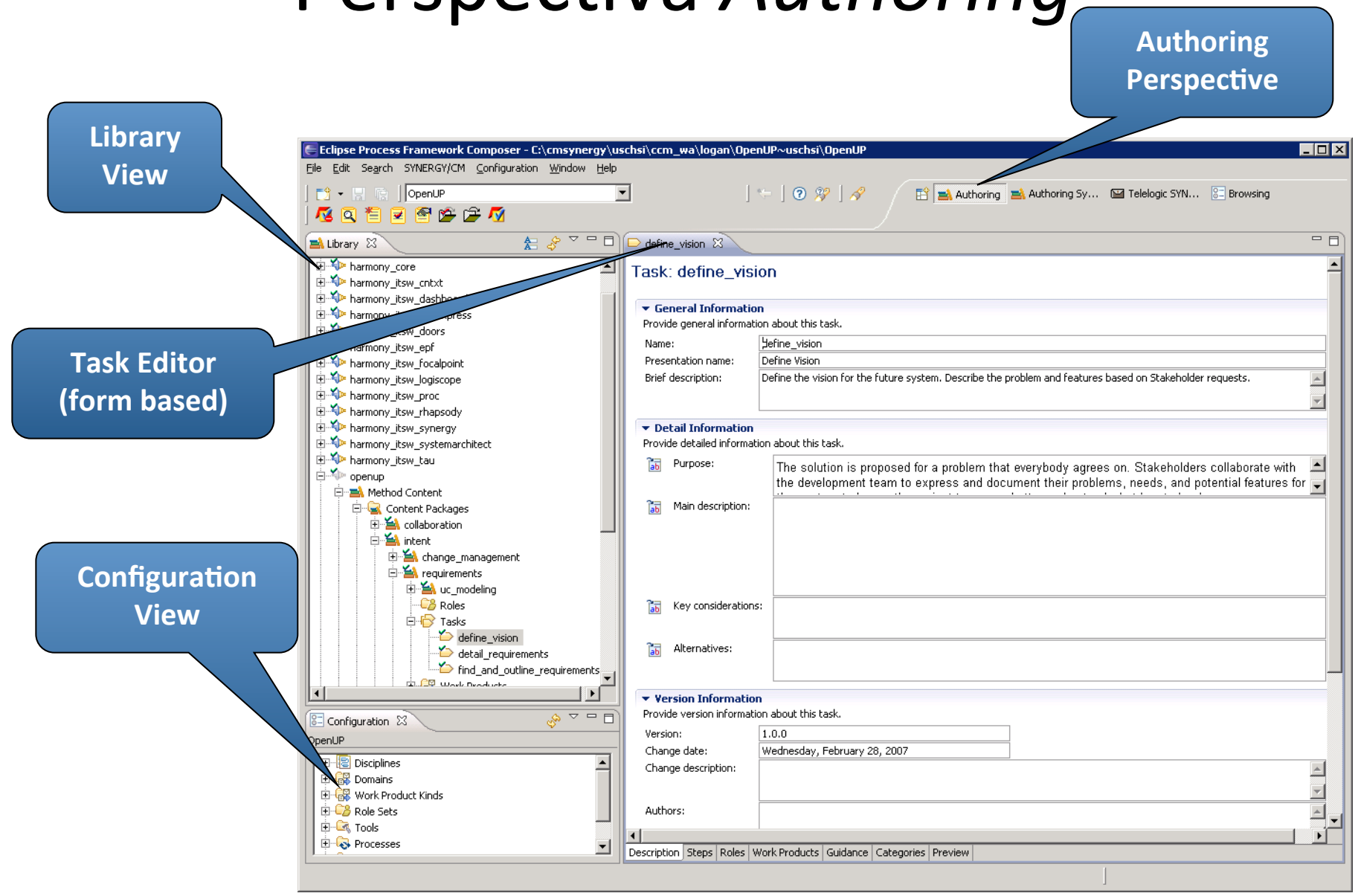
Ferramenta para autoria e publicação de processos de software

INTRODUÇÃO AO *EPF COMPOSER*

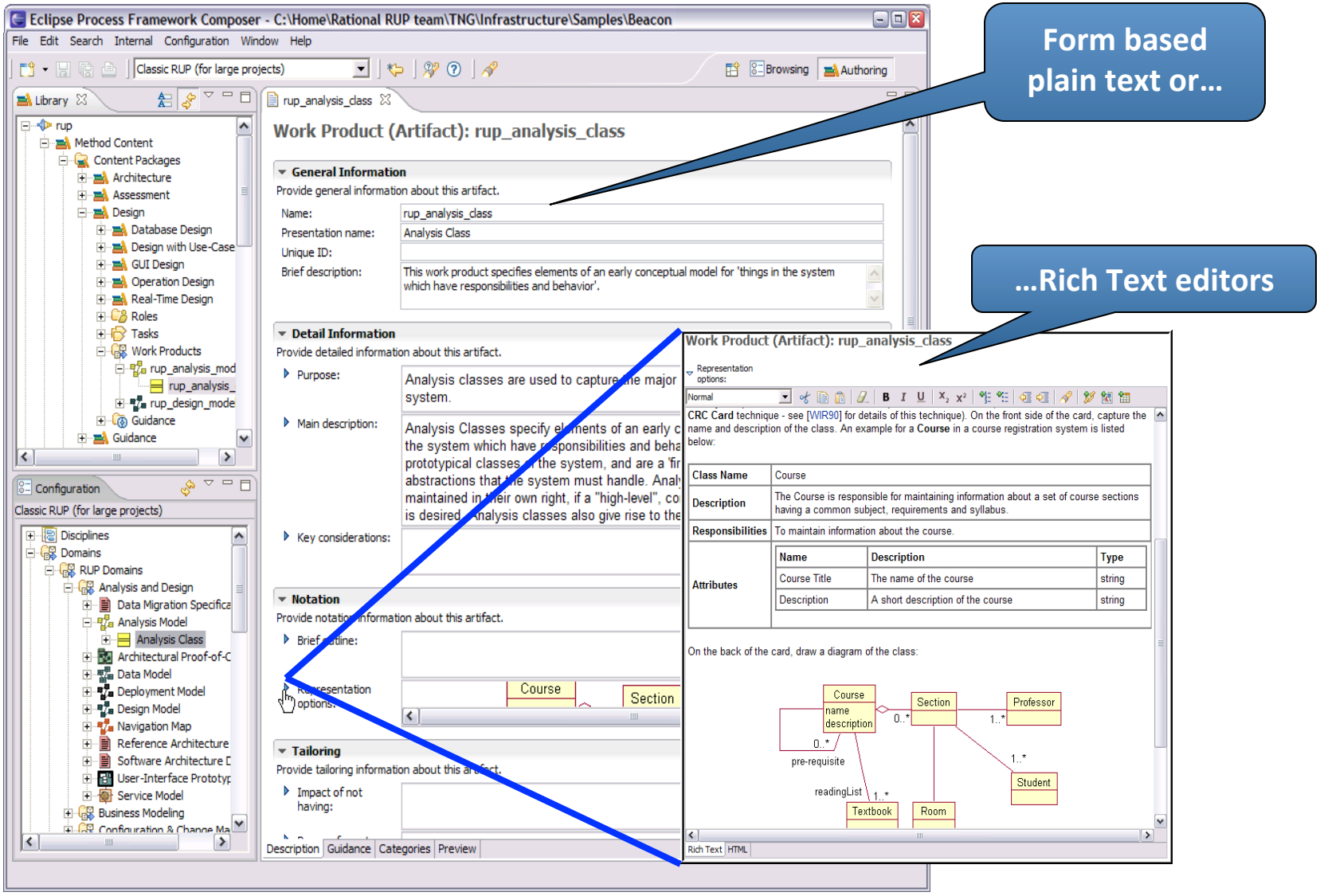
EPF Composer

- EPF Composer é desenvolvido sobre a plataforma Eclipse
- Suporta vários plug-ins do Eclipse
  - Ex.: Mylar
- Visões diferentes apresentam informações específicas
  - Ex.: a visão "Library" apresenta os plug-ins e seus conteúdos
- Perspectivas agrupam visões relacionadas
- Perspectivas padrão são:
  - *Authoring* → própria para a edição de conteúdos de método
  - *Browsing* → para visualizar uma prévia da publicação dos elementos de método e processo

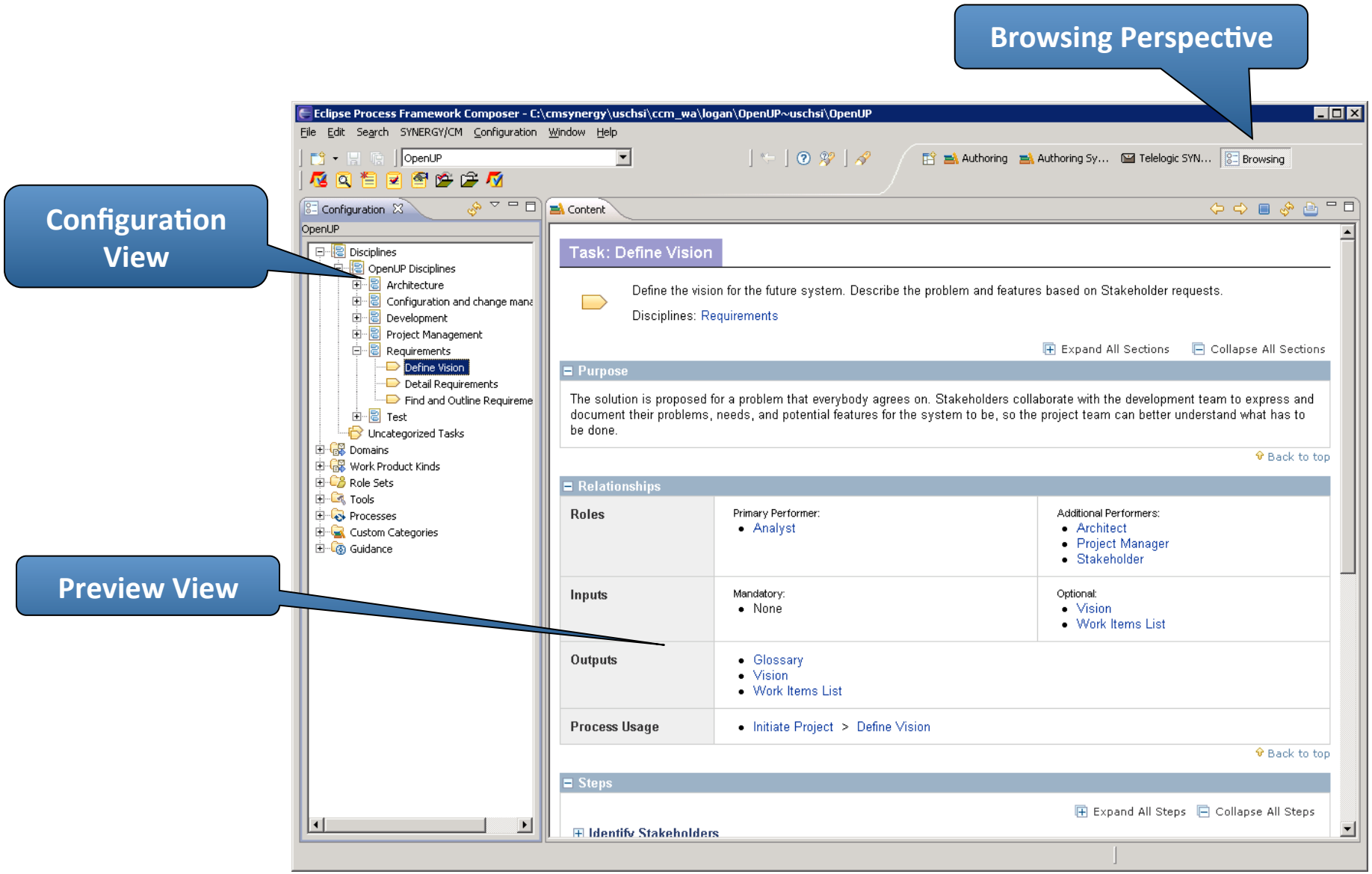
Perspectiva *Authoring*



Perspectiva *Authoring*

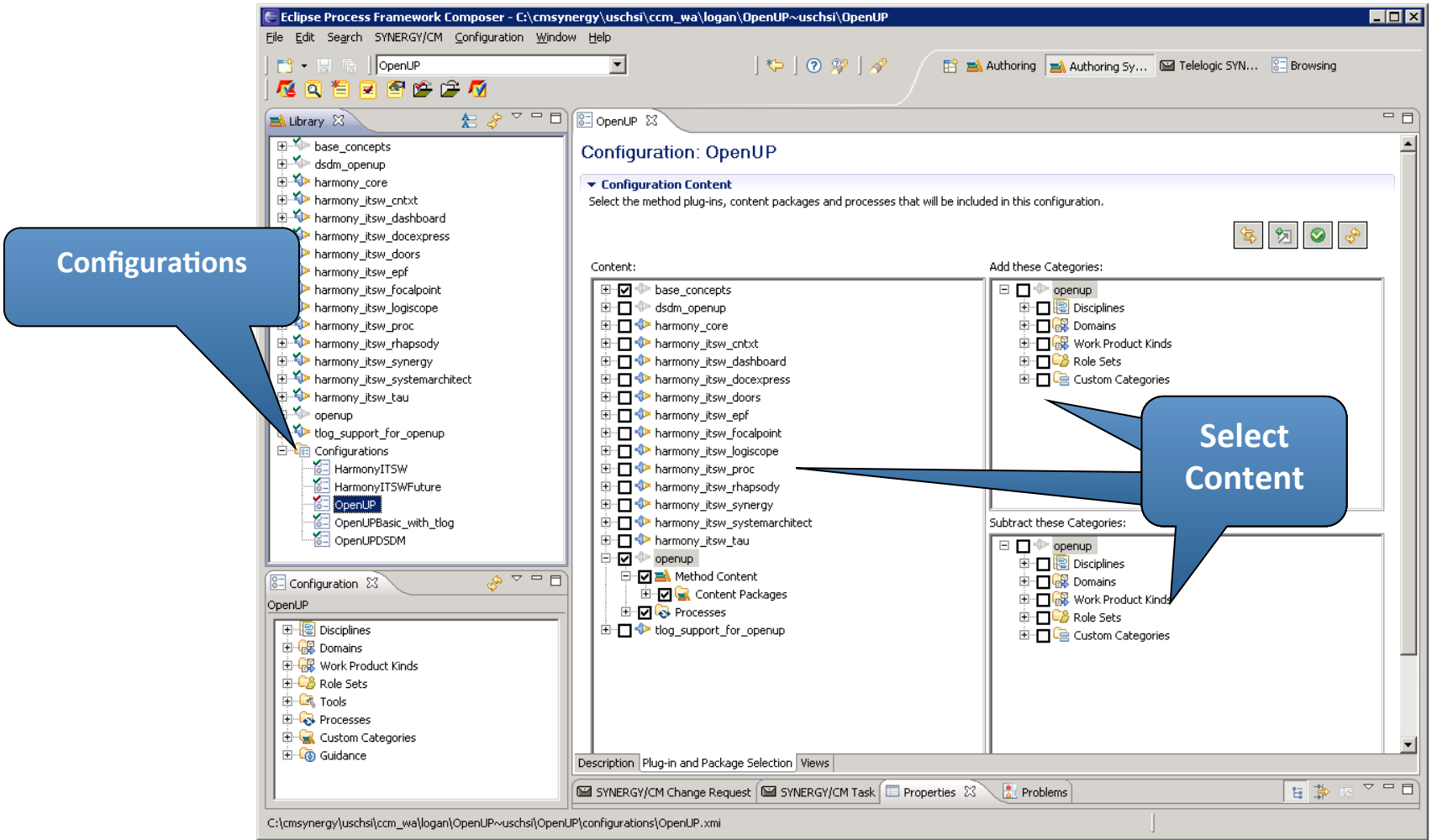


Perspectiva *Browsing*



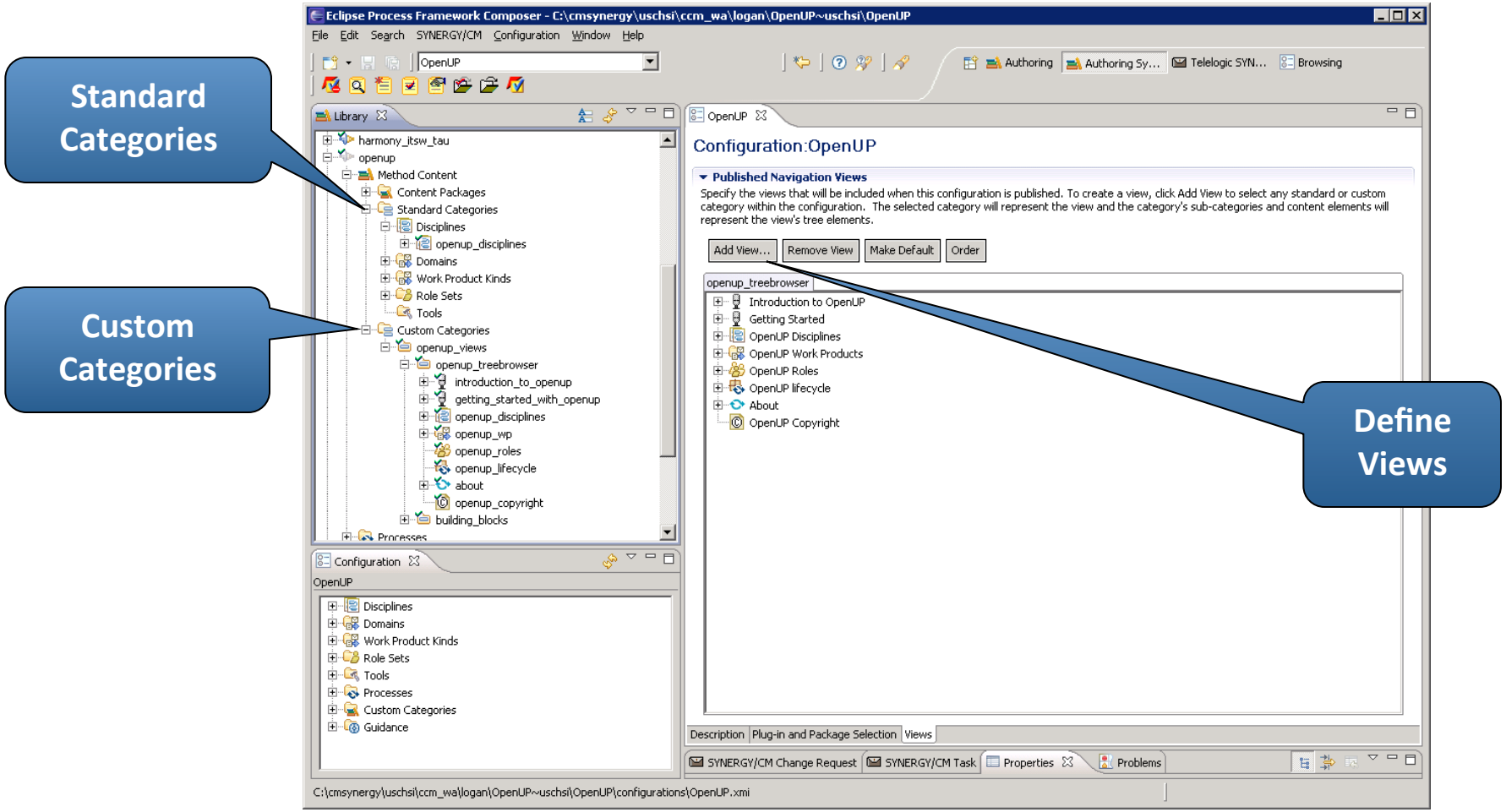
Configuração

- Seleção de um subconjunto da biblioteca de método para ser publicado em HTML ou exportado para MS Project ou XML



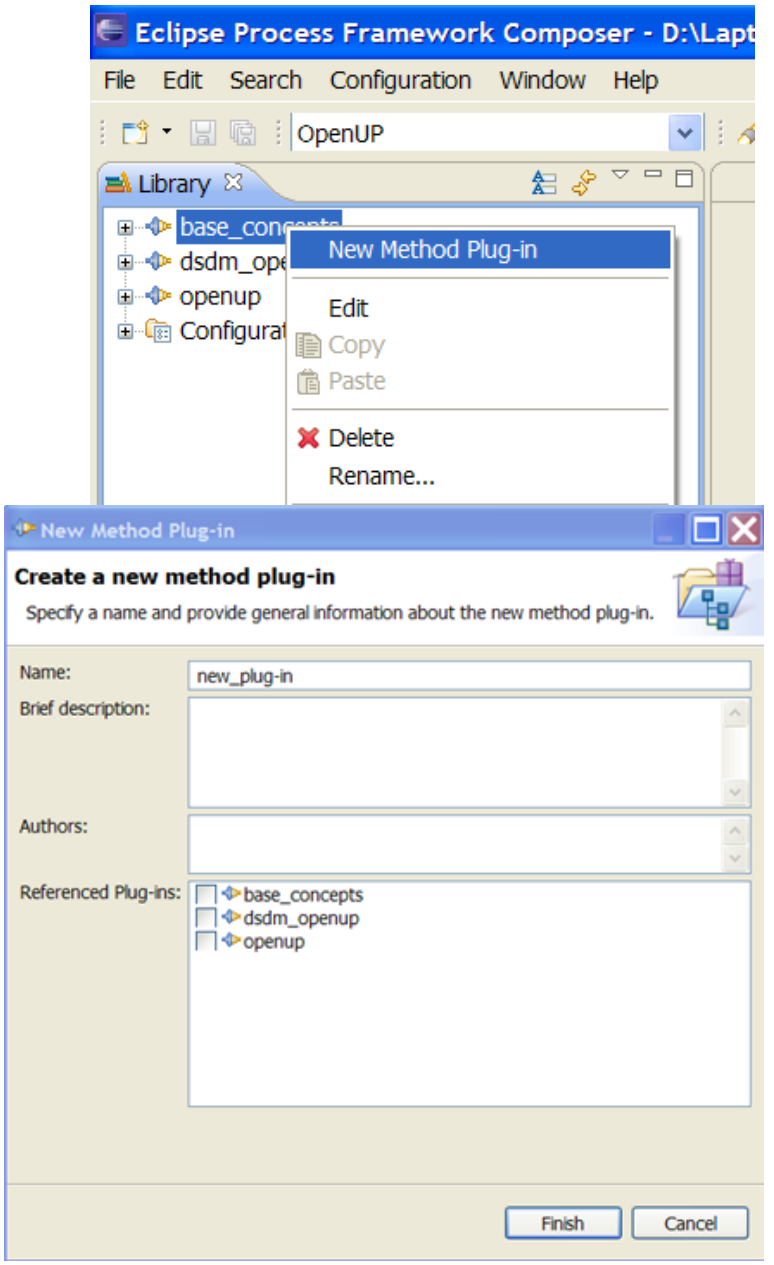
Configuração: Definição das Visões

- Categorias agrupam elementos relacionados
- Visões definidas pela seleção de categorias



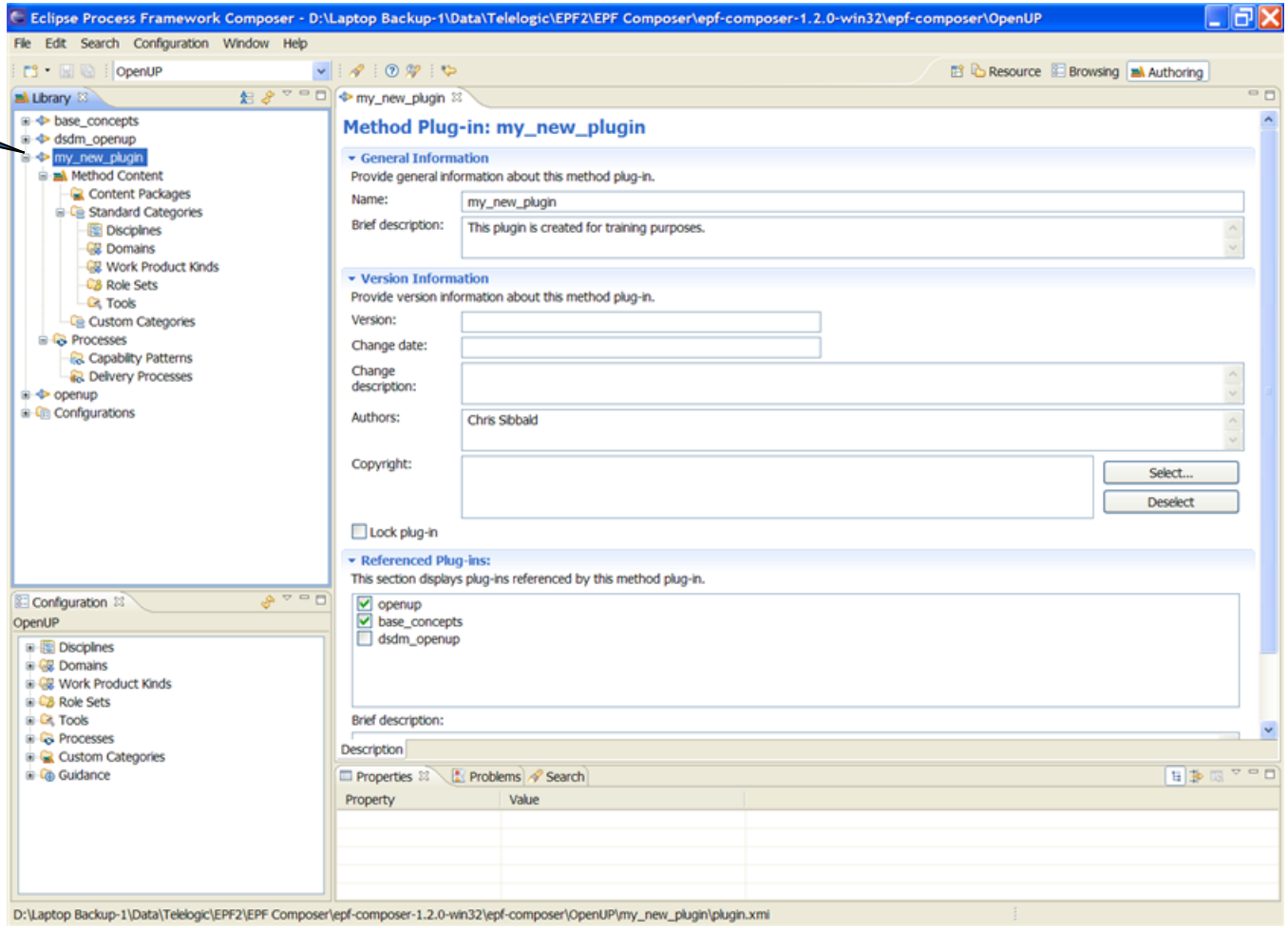
Criar um *Plugin*

- Clicando com o botão direito sobre um componente existente na biblioteca
  - Selecione “New Method Plug-in”
- Defina:
  - Nome (minúsculas, sem espaços)
  - Descrição (opcional)
  - Autor (opcional)
  - Plug-ins referenciados
- “Referenced Plug-ins” define a visibilidade a outros plug-ins



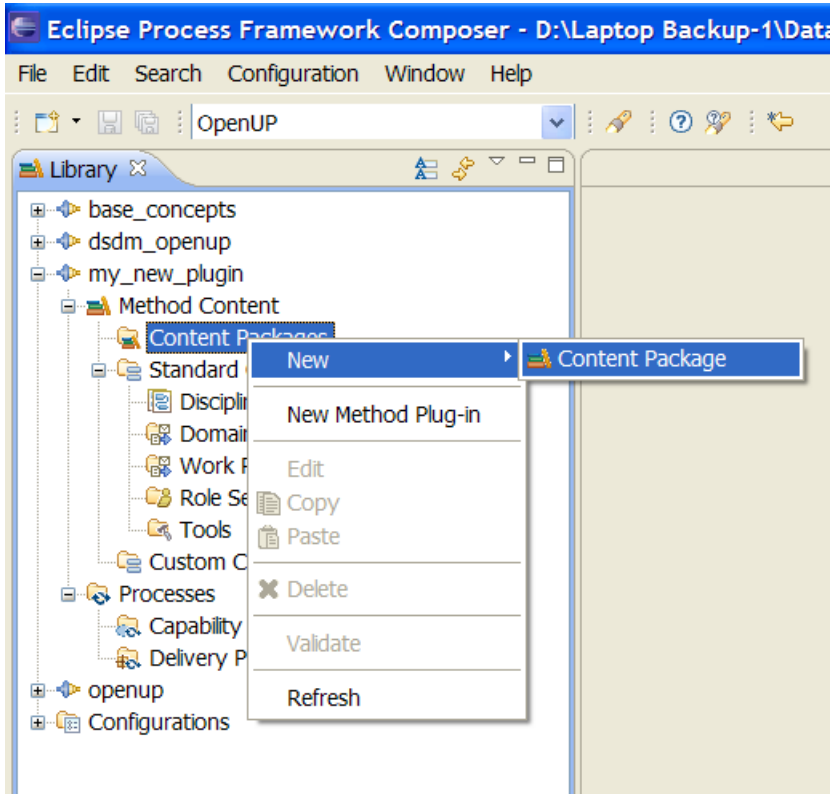
Criar um *Plugin*

New Plugin



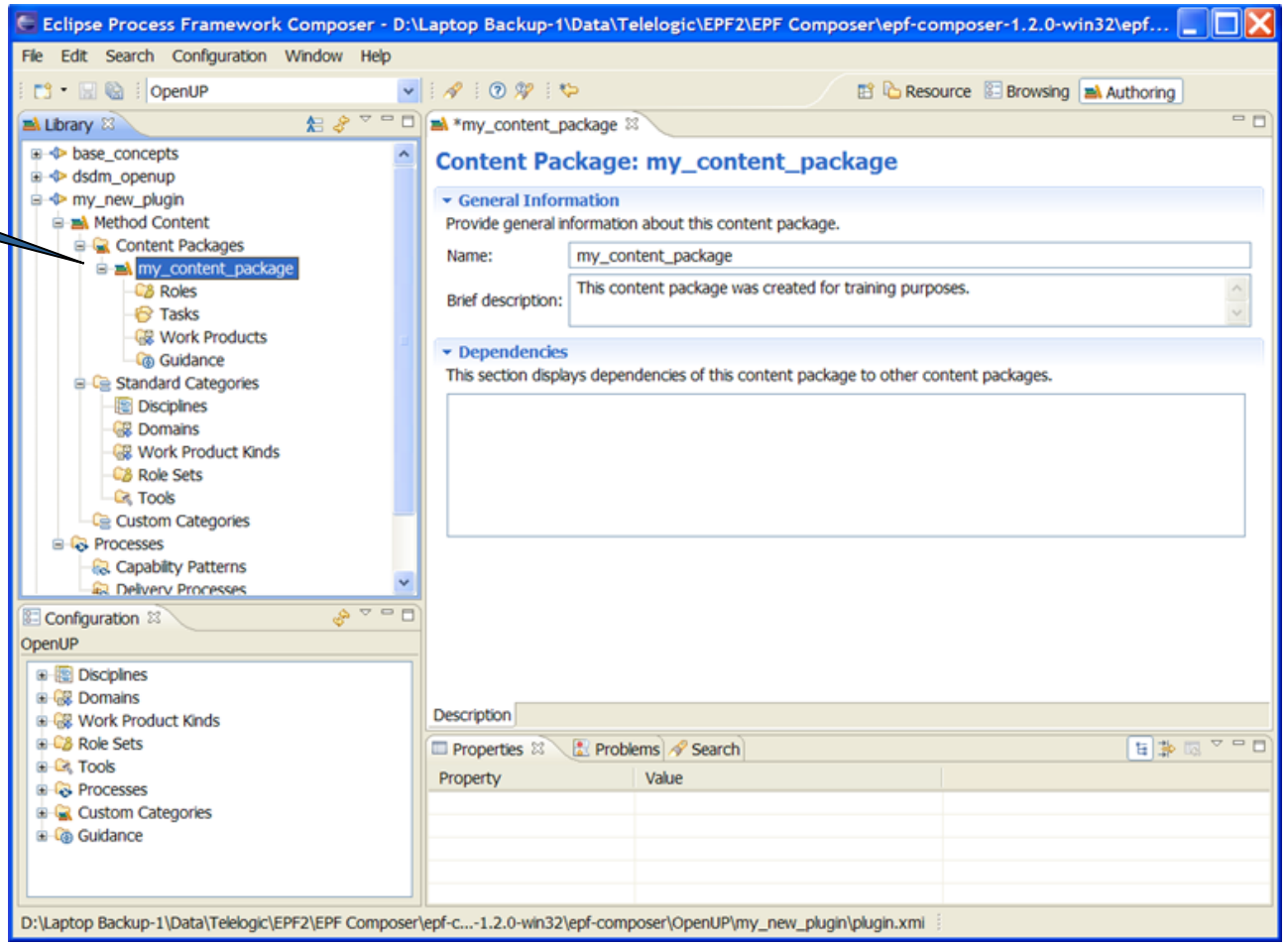
Criar um *Method Content Package*

- Clicar com o botão direito sobre “Content Packages”
  - Selecione “New Content Package”
- O novo pacote é criado
- Este é aberto no editor



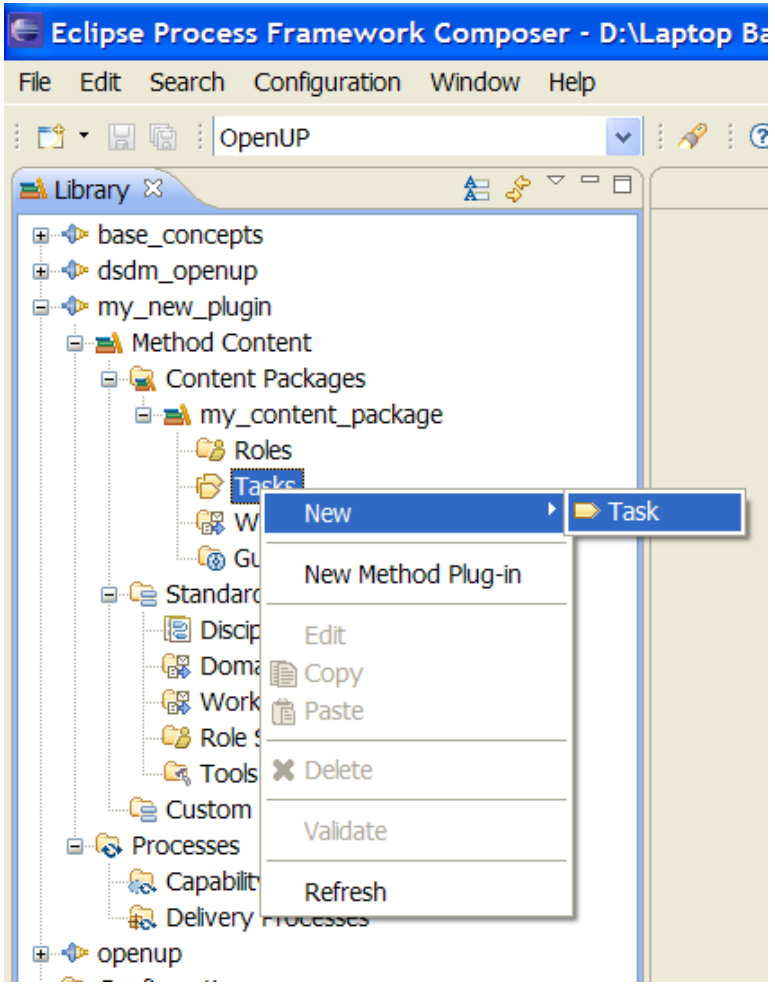
Criar um *Method Content Package*

New Content Package



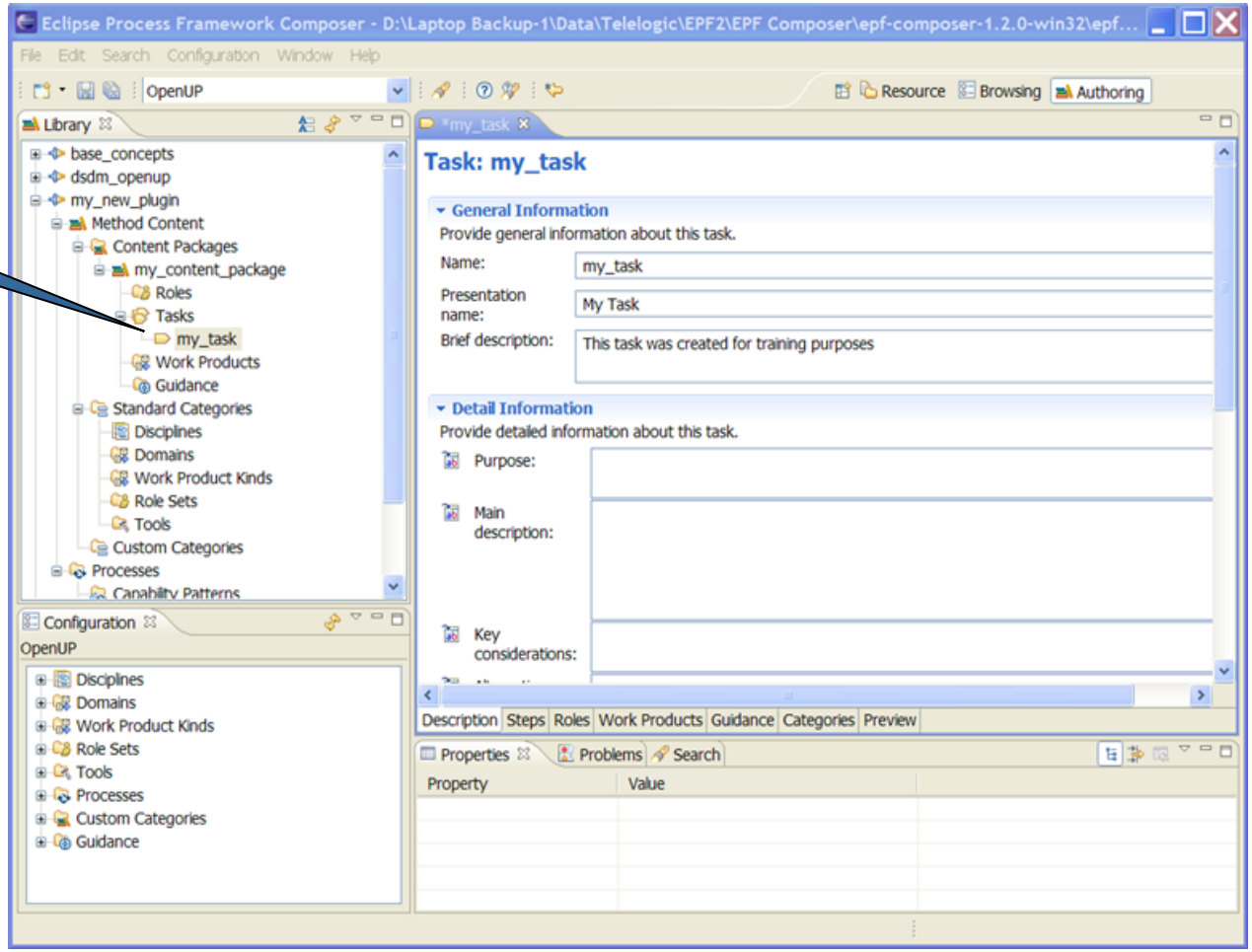
Criar uma Tarefa

- Clicar com o botão direito sobre “Tasks” no pacote de conteúdo desejado
  - Selecione “New Task”
- A nova tarefa é criada
- Este é aberto no editor



Criar uma Tarefa

New Task

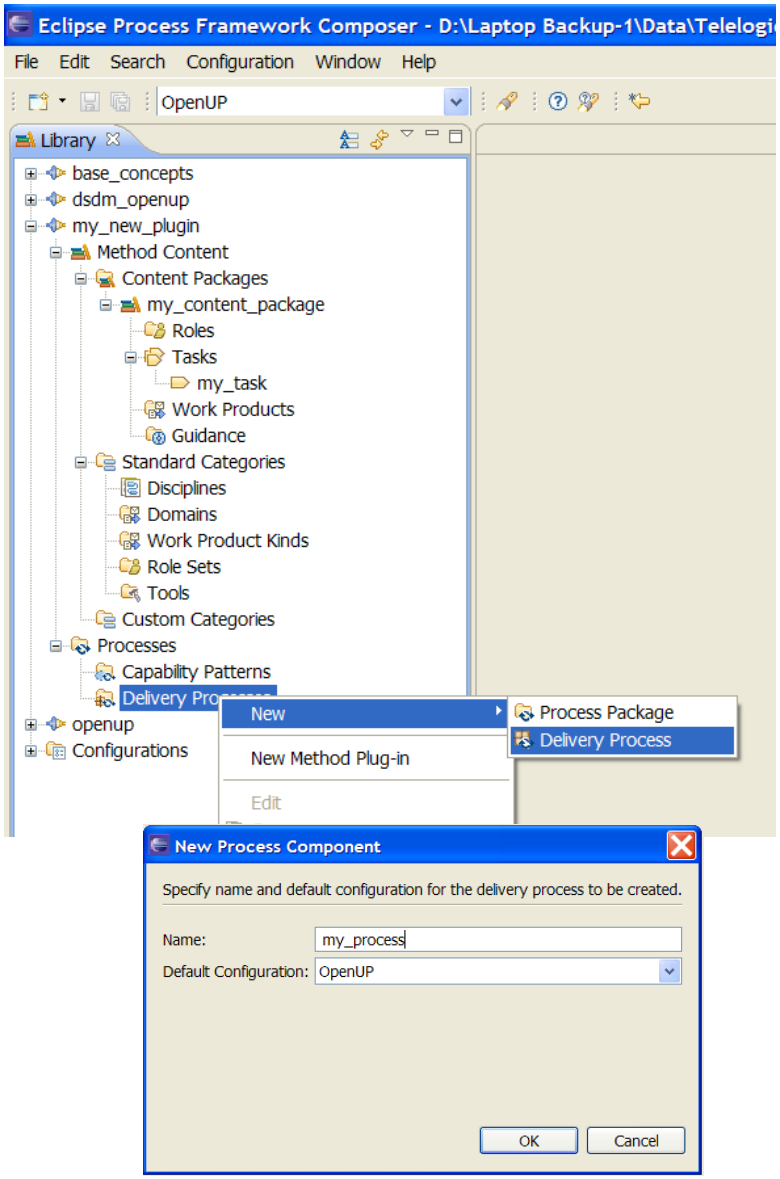


Editar uma Tarefa

- O editor de tarefas possui uma série de abas:
  - *Description* → definir uma visão geral da tarefa
  - *Steps* → definir os passos para a tarefa
  - *Roles* → definir os responsáveis pela tarefa
  - *Work Products* → definir artefatos de "entradas" e "saídas"
  - *Guidance* → associar elementos de orientação à tarefa
  - *Categories* → categorizar a tarefa
  - *Preview* → ver o resultado final
- Alguns campos possuem a capacidade de uma edição com mais recursos (*Ritch Text Editing*)

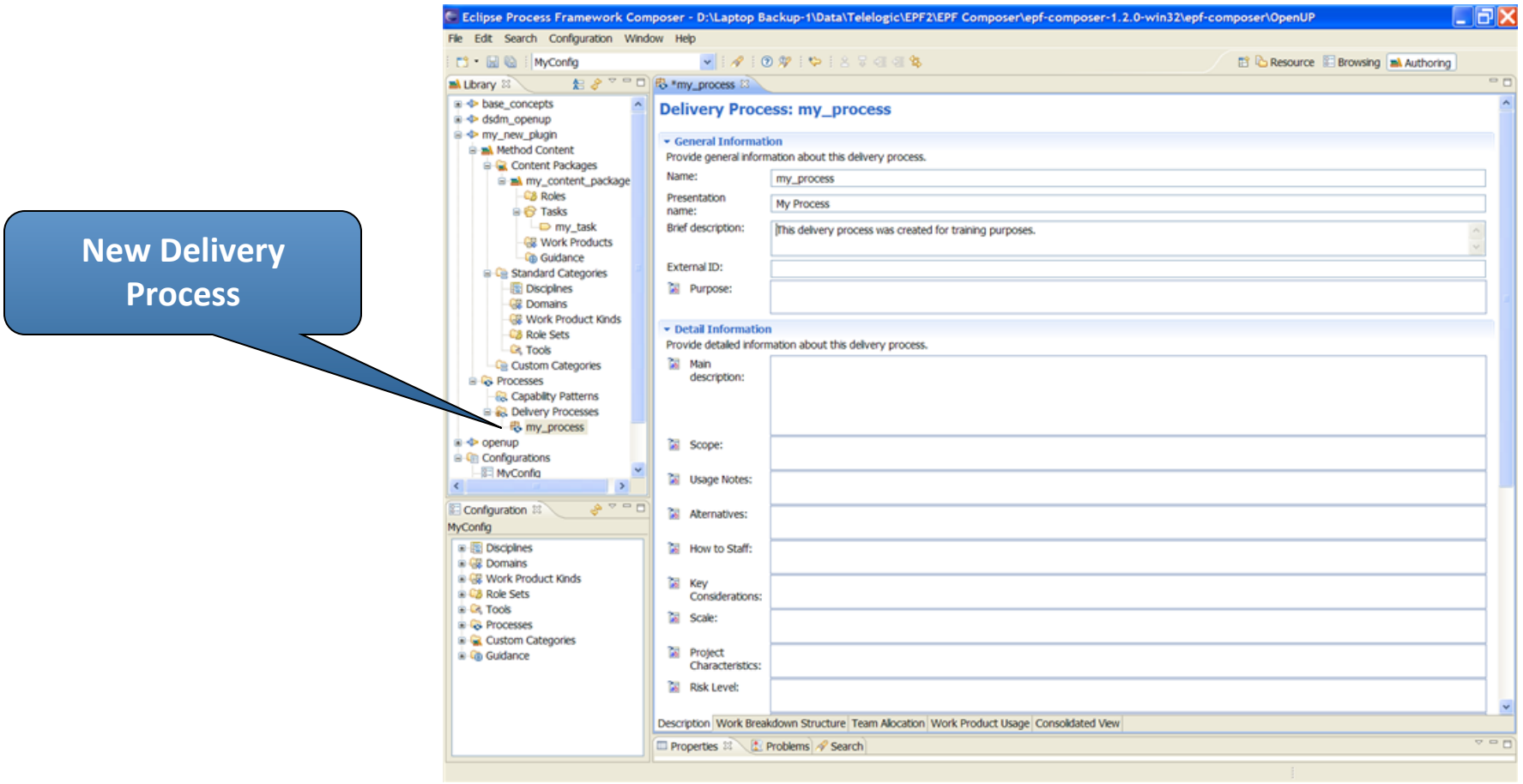
Criar um Delivery Process

- Clicar com o botão direito sobre “Delivery Process”
  - Selecione “New Delivery Process”
- Define-se:
  - Nome (minúsculas, sem espaço)
  - Configuração padrão



Criar um *Delivery Process*

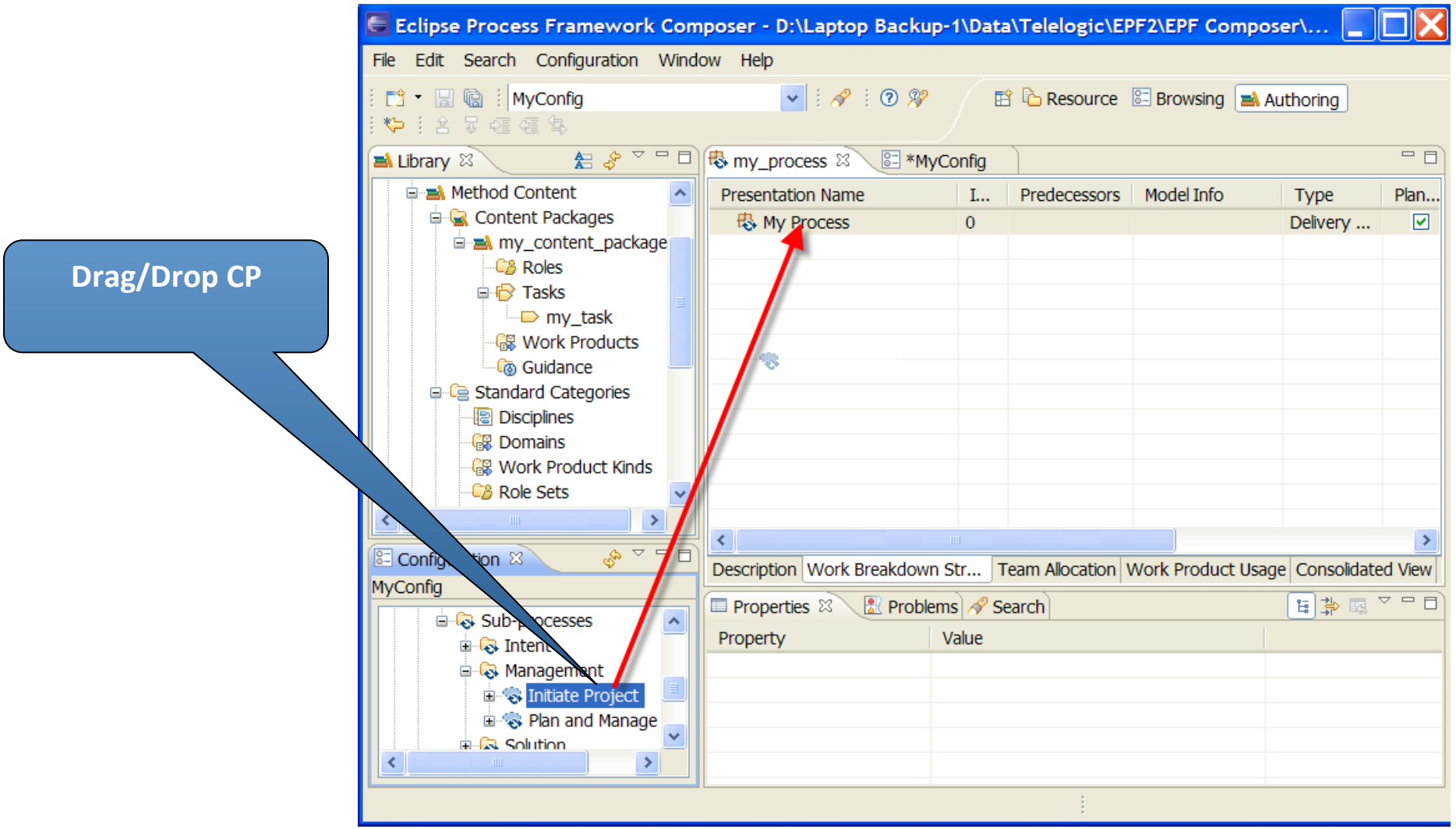
- O editor permite a alteração/atualização do conteúdo



Criar um *Delivery Process*

- O editor do Delivery Process possui uma série de abas:
  - *Description* → definir atributos gerais do processo
  - *WBS* → definir atividades e seus relacionamentos
  - *Team Allocation* → visualizar e editar papéis
  - *Work Products Usage* → visualizar e editar produtos de trabalho
  - *Consolidated View* → visão consolidada (preenchida automaticamente)

Criar um *Delivery Process*

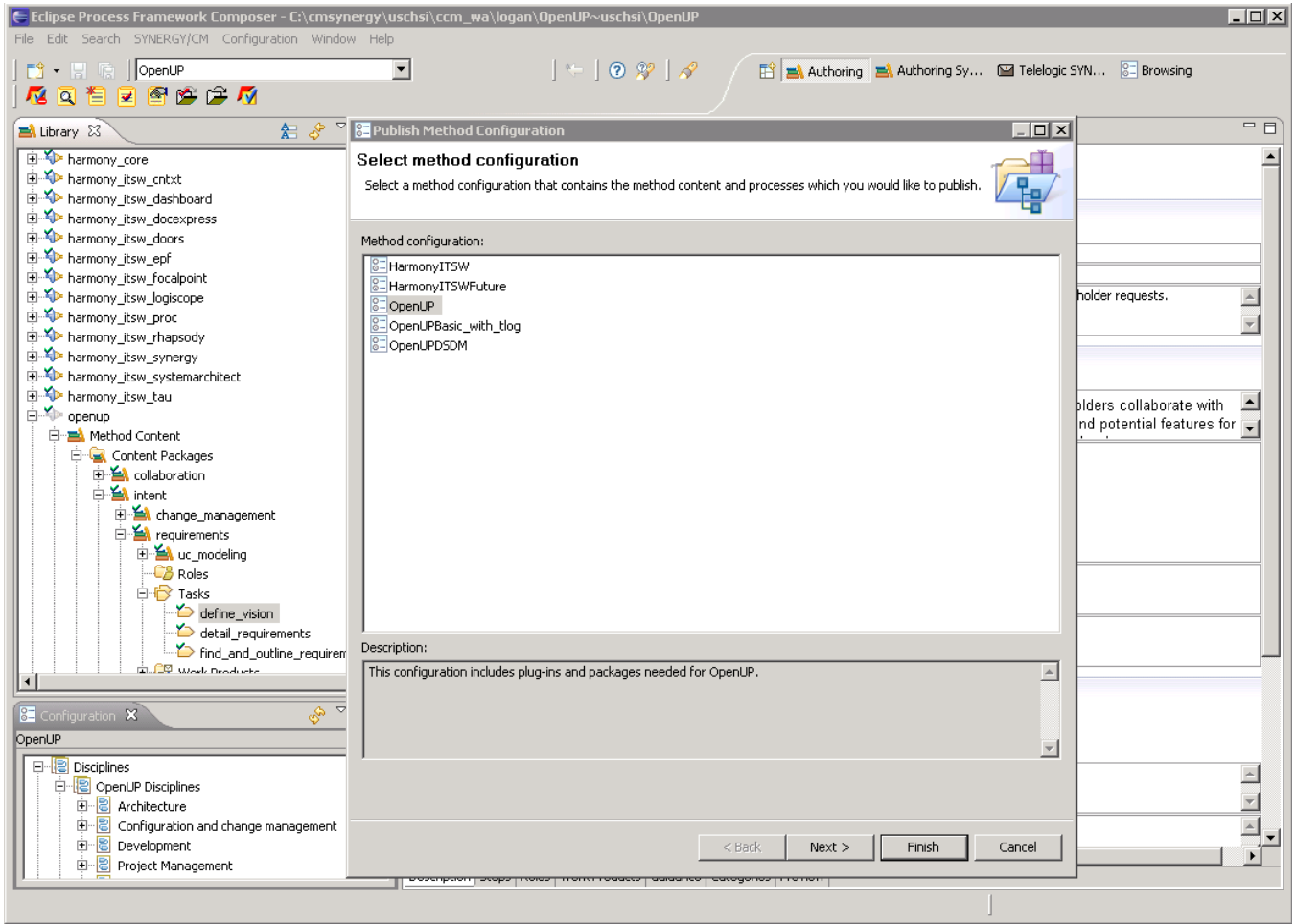


Criar um Delivery Process

- Ao adicionar um padrão de capacidade ao delivery process é possível:
  - *Extend* → irá manter um link ao CP, que caso seja atualizado, a mudança irá se refletir no *delivery process*
  - *Copy* → irá criar uma cópia local do CP, a qual não estará ligada ao CP original
  - *Deep Copy* → similar ao *copy*, mas é aplicado recursivamente aos sub-CPs
- A mais comum é a *Extend*

Publicação

- “Configuration | Publish” para iniciar o assistente de publicação



Website Resultado

http://usma-doorsdb/epf/openup/ - Microsoft Internet Explorer

File Edit View Favorites Tools Help

Eclipse Process Framework Composer

Glossary | Index | Feedback | About

Print

Where am I | Tree Sets

OpenUP

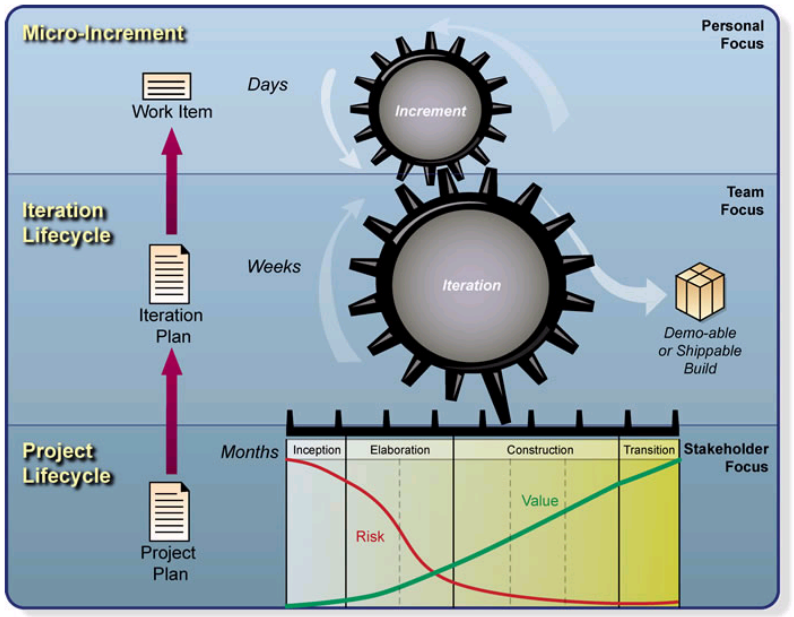
- Introduction to OpenUP
  - Core Principles
  - Minimal, Complete, and Ext
  - Iteration Lifecycle
  - Micro-Increments
  - Project Lifecycle
  - The OpenUP Family
  - Who Should Use OpenUP
- Getting Started
  - OpenUP Roadmap
  - Resources for Modifying Me
  - Resources for Contributing
- OpenUP Disciplines
  - Architecture
  - Configuration and Change
  - Development
  - Project Management
  - Requirements
  - Test
- OpenUP Work Products
  - Architecture
  - Development
  - Project Management
  - Requirements
  - Test
- OpenUP Roles
  - Analyst
  - Any Role
  - Architect
  - Developer
  - Project Manager
  - Stakeholder
  - Tester
- OpenUP lifecycle
- About
- OpenUP Copyright

Main Description

Getting Started | Core Principles | Roles | Work Products | Disciplines | Lifecycle

What is OpenUP?

OpenUP is a lean Unified Process that applies iterative and incremental approaches within a structured lifecycle. OpenUP embraces a pragmatic, agile philosophy that focuses on the collaborative nature of software development. It is a tools-agnostic, low-ceremony process that can be extended to address a broad variety of project types.



OpenUP layers: micro-increments, iteration lifecycle and project lifecycle

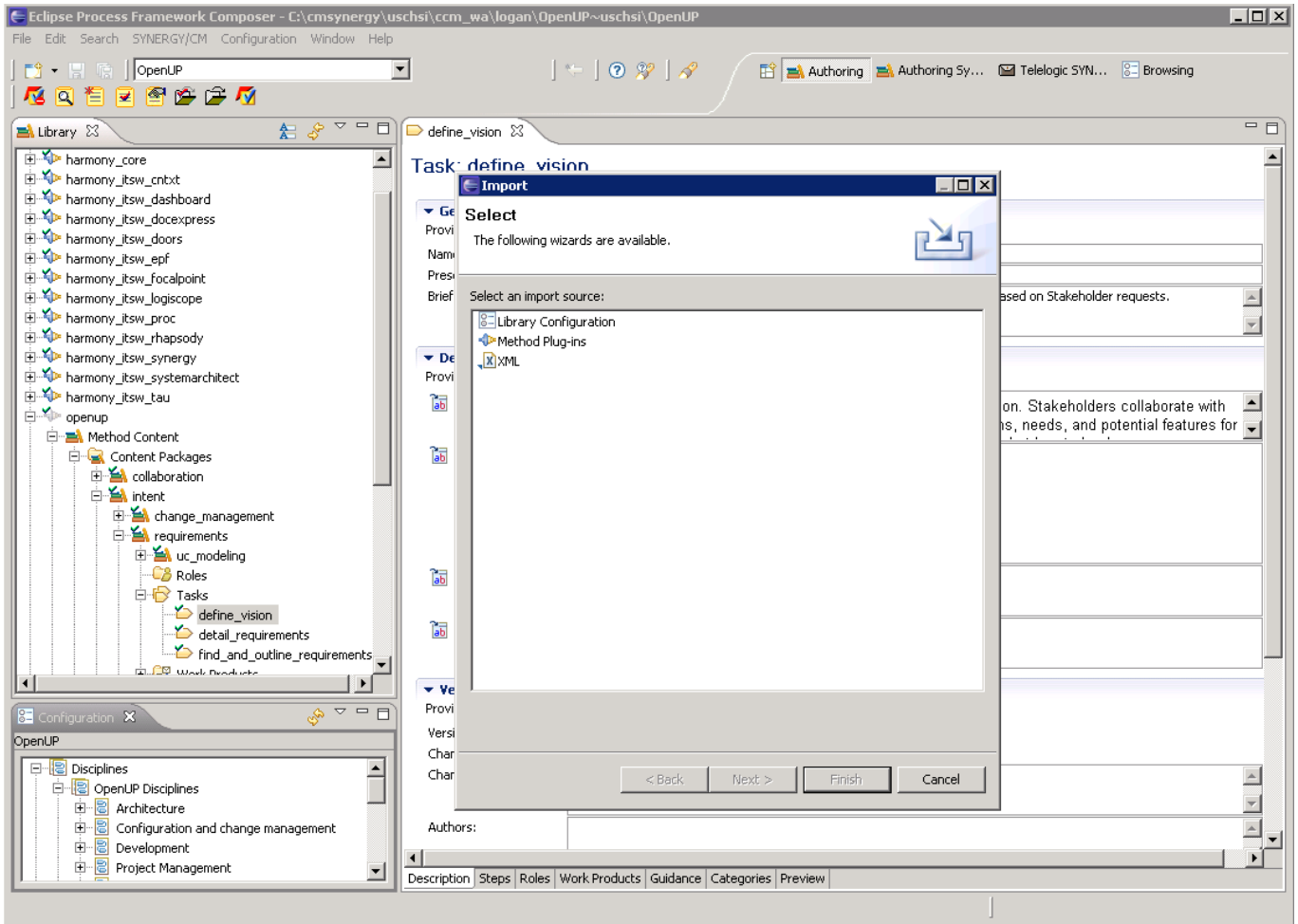
Personal effort on an OpenUP project is organized in **micro-increments**. These represent short units of work that produce a steady, measurable pace of project progress (typically measured in hours or a few days). The process applies intensive collaboration as the system is incrementally developed by a committed, self-organized team. These micro-increments provide an extremely

OpenUP lifecycle

Local intranet

Importação

- “File | Import” para iniciar o assistente de importação
- Pode ser importado uma Configuração, um plug-in, ou XML



Exportação

- “File | Export” para iniciar o assistente de exportação
- Pode exportar uma Configuração, plug-in, XML ou template MS Project