Osellus Inc.

Vivienne Suen

June 25, 2003

SPEM : Extending the UML into the Process Engineering Domain

Process Engineering

- While the importance of processes in software development is well-understood, there remains the question of *how* to leverage this understanding
  - – How do I define a process?
  - – How do I know my process is good?
  - – How do I know my process will work?
  - – etc.

Processes: Complex, Situational, Evolving

- Processes are complex – a good process is very specific, has distinct principles, and documents its own limitations
- Processes are situational – a minor change in a project's context can render a process ineffective
- Processes evolve – we need to be able to recognize change and adjust processes to fit

SPEM : The Software Process Engineering Metamodel

- SPEM is the OMG’s effort to leverage the expressiveness and popularity of modeling techniques to define and document software development processes

Uses for Process Models

- Accurate documentation of a development process (share with the whole team).
- Can be studied for enhancement, CMM assessment, etc.
- Can be easily tailored to suit different parts of the organization, or individual projects.
- The wonderful world of process automation!

What is SPEM?

- Grounded in UML concepts – already familiar to developers, architects, etc.
- Common notation and mapping of terminology
- Model interchangeability (XML)
- Eventually, integration with business process models and business rules definitions (~2 years)

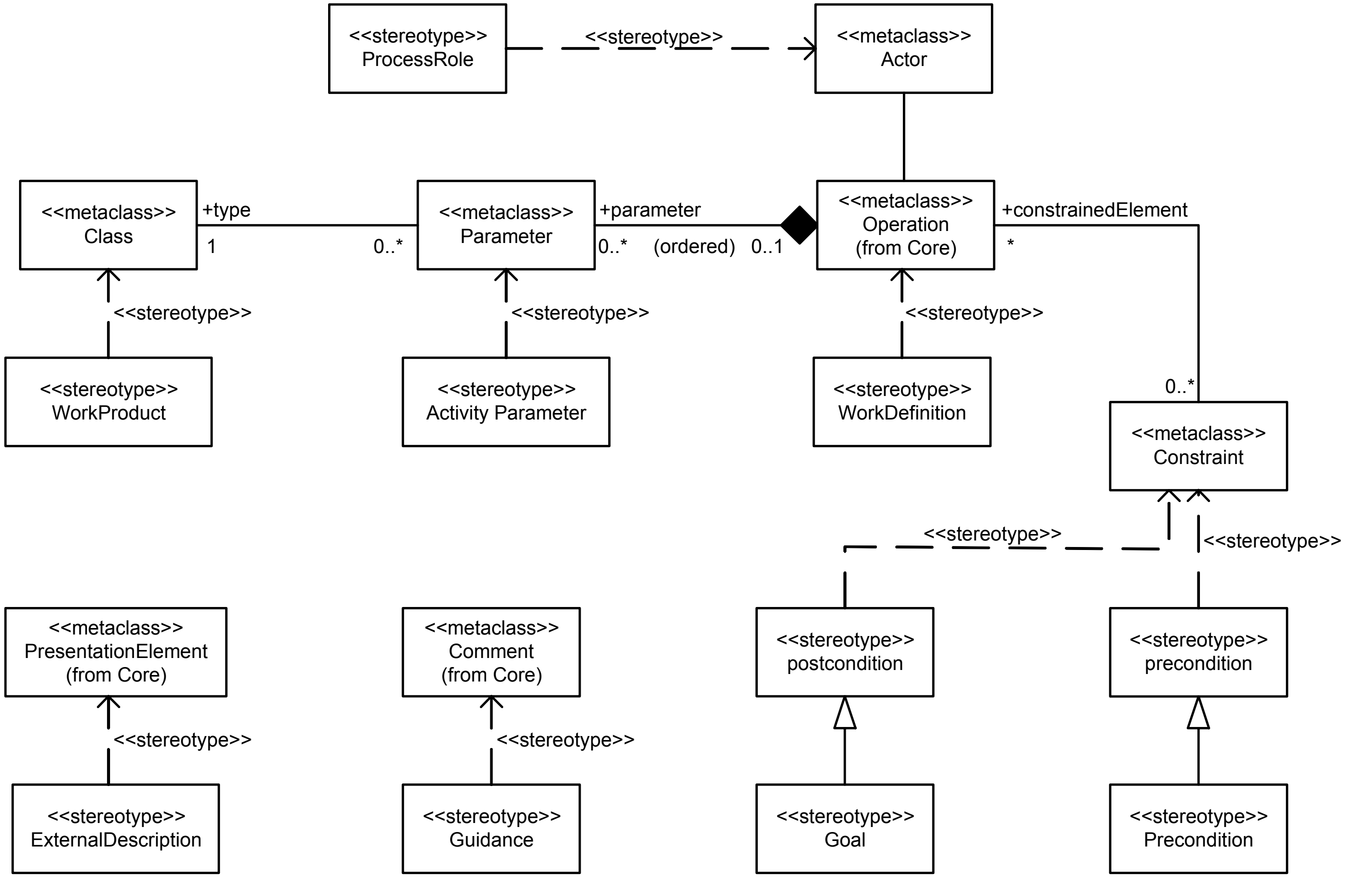
SPEM

- The SPEM Specification defines both a UML Profile and a pure standalone metamodel, along with an XMI DTD.
- UML Profiles are like “plug-ins” to UML-based modeling tools.
- The standalone metamodel does not rely on UML, but is based on basic OO/UML concepts.

The UML Profile

- Identified subset of UML 1.4 classes
- Mapping to UML base classes (plus emulation of SPEM attributes and associations)
- Additional constraints implied by the profiles
- Notational icons

Simple Mappings



<<stereotype>> ProcessRole
<<stereotype>>
<<metaclass>> Actor
<<metaclass>> Class
+type
1
0..*
<<metaclass>> Parameter
+parameter
0..*
(ordered)
0..1
<<metaclass>> Operation (from Core)
+constrainedElement
*
<<stereotype>>
<<stereotype>> WorkProduct
<<stereotype>>
<<stereotype>> Activity Parameter
<<stereotype>>
<<stereotype>> WorkDefinition
0..*
<<metaclass>> Constraint
<<stereotype>>
<<stereotype>>
<<metaclass>> PresentationElement (from Core)
<<stereotype>>
<<stereotype>> ExternalDescription
<<metaclass>> Comment (from Core)
<<stereotype>>
<<stereotype>> Guidance
<<stereotype>> postcondition
<<stereotype>> Goal
<<stereotype>> precondition
<<stereotype>> Precondition

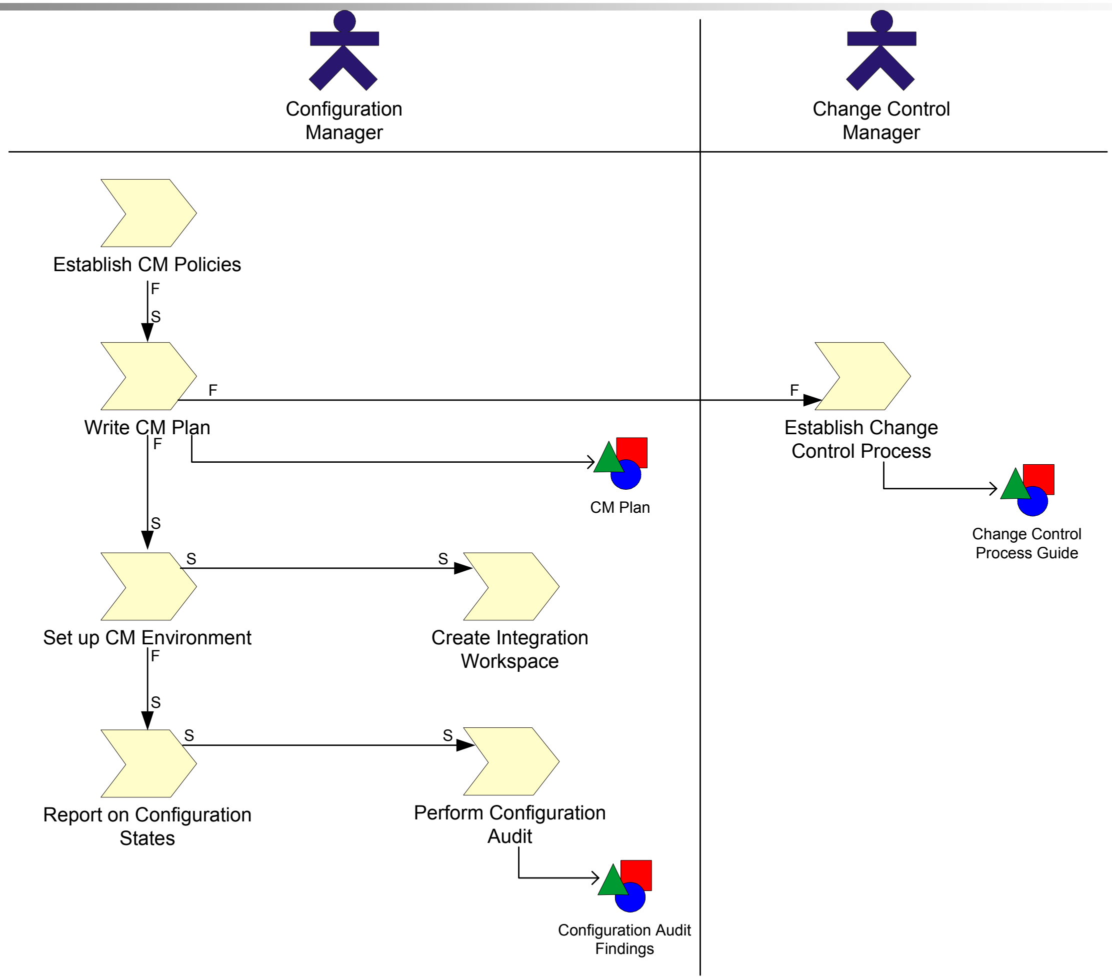
SPEM Metamodel

- Basic Elements (External Description, Guidance)
- Dependencies (Categorizes, Impacts, Import, Precedes, RefersTo, Trace)
- Process Structure (WorkProduct, WorkDefinition, Activity, Step, ProcessRole)
- Process Components (Package, ProcessComponent, Process, Discipline)
- Process Lifecycle (Lifecycle, Phase, Iteration, Precondition, Goal)

SPEM Modeling Concepts

- Work Breakdown Structure: modeled with Lifecycles, Phase, Iterations, WorkDefinitions, and Activities
- Artifacts and roles: modeled with WorkProducts and ProcessRoles, as inputs/outputs, and performers/assistants to Activities
- Packages – for modularity and reuse

An Example:


Configuration Manager
Change Control Manager
Establish CM Policies
F
S
Write CM Plan
F
F
Establish Change Control Process
F
CM Plan
Change Control Process Guide
S
S
S
Set up CM Environment
Create Integration Workspace
F
S
S
S
Report on Configuration States
Perform Configuration Audit
Configuration Audit Findings

SPEM 2.0

- It’s coming!
- The RFP is being worked on, and is planned for issue this year.

Thank you

For more information visit :

www.omg.org

www.osellus.com

The Software Process Engineering Metamodel