Process Modeling Principles

Marie-Pierre Gervais & …

UPMC-LIP6 & …
Context of this work

- The present courseware has been elaborated in the context of the MODELPLEX European IST FP6 project (http://www.modelplex.org/).
- Co-funded by the European Commission, the MODELPLEX project involves 21 partners from 8 different countries.
- MODELPLEX aims at defining and developing a coherent infrastructure specifically for the application of MDE to the development and subsequent management of complex systems within a variety of industrial domains.
- To achieve the goal of large-scale adoption of MDE, MODELPLEX promotes the idea of a collaborative development of courseware dedicated to this domain.
- The MDE courseware provided here with the status of open-source software is produced under the EPL 1.0 license.
Outline

- Basic concepts of process modeling
- Comparison of process technology domains
- Survey of UML-Based software process modeling languages
I - Basic concepts of process modeling
[Bendraou07a]

- Process
- Process Model
- Process Modeling Language
Process: a general definition

  - "A process uses resources to transform inputs into outputs. In every case, inputs are turned into outputs because some kind of work, activity or function is carried out"

- The term “process” is used in various domains (and by various communities in each domain)
  - Definitions may differ slightly or may be completely different from one domain into another
    - e.g. Computer Science domain Vs. Chemical Industry domain

- We focus on three communities of the computer and information science domain
  - Workflow management community
  - Business Process Management community
  - Software Process Engineering community
Process definition in Workflow management community

- Workflow Management Coalition, Terminology and Glossary, WFMC-TC-1011, 1999 [WFMC99]

- "a process is a formalized view of a business process, represented as a coordinated (parallel and/or serial) set of process activities that are connected in order to achieve a common goal"

- "a business process is a set of one or more linked procedures or activities, which collectively realize a business objective or policy goal, normally within the context of an organizational structure defining functional roles and relationships"
Process definition in Business process management community

- Many definitions, e.g. [Bastida05], [Davenport93], [Johansson93], [OMG06]

- [Davenport93]: most representative definition (in our view)
  - “a business process is a structured, measured set of activities designed to produce a specific output for a particular customer or market. It implies strong emphasis on how work is done within an organization, in contrast to a product focus’s emphasis on what. A process is thus a specific ordering of work activities across time and space, with a beginning and an end, and clearly defined inputs and outputs: a structure for process actions”
Process definition in Software Process Engineering community

- Many definitions, e.g. [Lonchamp93], [Humphrey89], [Sommerville07], [Fuggetta00], [OMG05], [ISO06]

- [Lonchamp93]: most representative definition
  - “a software process is the set of partially ordered process steps, with sets of related artifacts, human and computerized resources, organizational structures and constraints, intended to produce and maintain the requested software deliverables”

- [Humphrey89]: most cited in literature
  - “a software engineering process is the total set of software engineering activities needed to transform a user’s requirements into software”
Process definition: Summary

- **Similarities between the three communities**
  - Process: (roughly) a set of activities required to transform inputs into outputs

- **Differences between the three communities**
  - The objectives the process tends to attain
    - e.g. a reliable product or service, an added value in terms of benefits or customer satisfaction, a rigorous agent coordination, etc.
  - The means and vocabulary are proper to each process domain
  - The result of applying the process
Process Models

- [Curtis92]
  - “a process model is an abstract description of an actual or proposed process that represents selected process elements that are considered important to the purpose of the model and can be enacted by a human or a machine”

- [Scacchi01]
  - A software process model represents a networked sequence of activities, objects, transformations and events that embody strategies for accomplishing software evolution

- Constituents of software process models
  - Activity, artifact, role, human, tool
Process Models - Purposes

- Facilitate human understanding and communication
- Facilitate process reuse and repeatability
- Support of process analysis and improvement
- Support process management (process’s status)
- A basis automate agents coordination and guidance
- A basis to automate execution support of the process (automation of repetitive and non interactive steps)
Process Models - views (1)

• One view cannot capture all process concerns!
• The **activity** view: focuses on the types, structure and properties of the activities in the process and their dependencies.
  • Use: Process understanding and reasoning, scheduling purposes and monitoring, possibly a basis for process automation, etc.;
• The **product** view: describes the types, structure and properties of process’s artifacts.
  • Use: identification of process’s deliverables, relations and dependencies between process’s artifacts, the impact of an artifact changes on the other artifacts, etc.;
Process Models - views (2)

- The resource view: describes the resources (human or physical) either needed from or supplied to the process, which is relevant from a managerial perspective;
  - Use: project planning, resource allocations, etc.

- The role view: describes a particular set of resources, namely skills that performers supply and the responsibilities they accept.
  - Use: Process planning and management, access rights affectations, etc..
Process Modeling Languages

- [Zamli01]
  - “a process modeling language is a language used to express process models”

Process modeling language (PML) requirements

- From the consideration: The PML is a modeling language
  - Expressiveness (semantic richness), understandability, modularity
    - [Conradi95], [Kellner89]

- From the consideration: The PML is a programming language
  - Executability, formality
    - [Osterweil87]
II - Comparison of process technology domains
[Bendraou07b]

- Process definition and goal of modeling processes
- Process composition
- Domain and process characteristics
- Process model elements and domain’s context and scope
- Relationship with other domains
# Process definition and goal of modeling processes

<table>
<thead>
<tr>
<th>Definition of the term Process</th>
<th>Software Process Engineering (SPE)</th>
<th>Business Process Management (BPM)</th>
<th>Workflow Management (WfM)</th>
</tr>
</thead>
<tbody>
<tr>
<td>The set of partially ordered process steps, with sets of related artifacts, human and computerized resources, organizational structures and constraints, intended to produce and maintain the requested software deliverables [Lonchamp93]</td>
<td>A specific ordering of work activities across time and space, with a beginning and an end, and clearly defined inputs and outputs. Processes are the structure by which an organization does what is necessary to produce value for its customers [Davenport93]</td>
<td>A formalized view of a business process, represented as a coordinated (parallel and/or serial) set of process activities that are connected in order to achieve a common goal [WFMC99]</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Goals of Modeling Process</th>
<th>Primary Goal</th>
<th>Secondary Goals</th>
<th>Primary Goal</th>
<th>Secondary Goals</th>
<th>Primary Goal</th>
<th>Secondary Goals</th>
</tr>
</thead>
<tbody>
<tr>
<td>To facilitate and to support the development of high-quality software more quickly and at lower cost</td>
<td>Support for human understanding, analysis, measurement, simulation, execution, and improvement</td>
<td>Creation/addition of Value for the customer and for the organisation</td>
<td>Automatic routing of artifacts across process activities and participants</td>
<td>Saving of time by automatic task and artifacts affectations, detection of process bottlenecks</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Process composition

<table>
<thead>
<tr>
<th>Process Composition (Sub-Processes or Phases)</th>
<th>Software Process Engineering (SPE)</th>
<th>Business Process Management (BPM)</th>
<th>Workflow Management (WFM)</th>
</tr>
</thead>
<tbody>
<tr>
<td>The Software Production Process: represents the process being actually performed by software developers and tools</td>
<td>The Core Process: the primary activities that converted raw materials into finished products and sold and delivered them</td>
<td>The Support Process: the activities of process automation, managing resource and process, analysis and improvement</td>
<td>There is no firm consensus in the literature on process phases in the Workflow domain. The WfMC define two phases: Process Definition Phase: designates the time period when manual and/or automated (workflow) descriptions of a process are defined and/or modified electronically Process Execution Phase: the time period during which the process is operational, with process instances being created and managed</td>
</tr>
<tr>
<td>The Meta-Process: consists of the activities of modeling the process, managing the process, support for its execution and improvement</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# Domain and process characteristics

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>- complex</td>
<td>- complex</td>
<td>- may cross functional/organisational boundaries</td>
<td></td>
</tr>
<tr>
<td>- exception driven</td>
<td>- exception driven</td>
<td>- High degree of natural concurrency—many interrelated tasks are running at any given point of the business process</td>
<td></td>
</tr>
<tr>
<td>- highly unpredictable since they depend too much on too many people and circumstances</td>
<td>- highly unpredictable since they depend too much on too many people and circumstances</td>
<td>- There is a requirement to monitor and manage the overall business process</td>
<td></td>
</tr>
<tr>
<td>- Not all activities are supported by automated tools</td>
<td>- Not all activities are supported by automated tools</td>
<td>- complex, highly dynamic and unpredictable</td>
<td></td>
</tr>
<tr>
<td>- They depend on communication, coordination and cooperation within a predefined framework</td>
<td>- They depend on communication, coordination and cooperation within a predefined framework</td>
<td>- often long running</td>
<td></td>
</tr>
<tr>
<td>- Their success depends on user involvement and the coordination of many roles</td>
<td>- Their success depends on user involvement and the coordination of many roles</td>
<td>- need user interactions</td>
<td></td>
</tr>
<tr>
<td>- They may take a long time and are subject to changes during this time</td>
<td>- They may take a long time and are subject to changes during this time</td>
<td>- may need to migrate in the event of hardware failure or for performance</td>
<td></td>
</tr>
<tr>
<td>- stateful</td>
<td>- stateful</td>
<td>- stateful</td>
<td></td>
</tr>
<tr>
<td>- they have customers (internal or external)</td>
<td>- they have customers (internal or external)</td>
<td>- they have customers (internal or external)</td>
<td></td>
</tr>
<tr>
<td>- Workflow systems are suitable for supporting rigidly structured, well-defined and repeatable business processes</td>
<td>- Workflow systems are suitable for supporting rigidly structured, well-defined and repeatable business processes</td>
<td>- Workflow systems are suitable for supporting rigidly structured, well-defined and repeatable business processes</td>
<td></td>
</tr>
<tr>
<td>- Processes are more often implemented in terms of proprietary code and procedures</td>
<td>- Processes are more often implemented in terms of proprietary code and procedures</td>
<td>- Processes are more often implemented in terms of proprietary code and procedures</td>
<td></td>
</tr>
<tr>
<td>- Business rules are hard-coded. Once the process execution is launched, it is not possible to modify them</td>
<td>- Business rules are hard-coded. Once the process execution is launched, it is not possible to modify them</td>
<td>- Business rules are hard-coded. Once the process execution is launched, it is not possible to modify them</td>
<td></td>
</tr>
<tr>
<td>- Workflow systems are tightly coupled through customized APIs with software and applications used during the process</td>
<td>- Workflow systems are tightly coupled through customized APIs with software and applications used during the process</td>
<td>- Workflow systems are tightly coupled through customized APIs with software and applications used during the process</td>
<td></td>
</tr>
<tr>
<td>- Workflows are limited in modeling all aspects of business processes</td>
<td>- Workflows are limited in modeling all aspects of business processes</td>
<td>- Workflows are limited in modeling all aspects of business processes</td>
<td></td>
</tr>
<tr>
<td>- Not all workflow systems provide graphical representations</td>
<td>- Not all workflow systems provide graphical representations</td>
<td>- Not all workflow systems provide graphical representations</td>
<td></td>
</tr>
</tbody>
</table>
## Process model elements and domain’s context and scope

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Activity, Artifact, Role, Tool and Agent</td>
<td>Deliverable, Constraint, Milestone, Guidance, Team, Phase, Iteration, Lifecycle, etc.</td>
<td>Organizational Unit or Entity, Customer, Software, Goal, Process Owner, Service, Business Rule, Event, Message, Condition, Date, Transaction, Sub-Process, Time, etc.</td>
<td>Activity, Artifact, Role, Tool and Agent</td>
<td>Activity, Artifact, Role, Tool and Agent</td>
<td>Work Item, Work List, Task (automated activity), Instance (of process or activity), Deadline, Procedure, Rule, Application, Event, etc.</td>
</tr>
</tbody>
</table>

### Domain’s context and scope
- **Organisations having as main business the development and maintenance of software.**
- **In addition, organisations that tend to retain their expertise and strategic business by developing their own software.**
- **The scope of the process is the organisation.**
- **Any organisation, whatever its business, that wants to create/add value to its customers.**
- **Business processes span many organization's units as well as many other organizations.**
- **Workflows are often Organisation/Department specific.**
## Relationship with other domains

<table>
<thead>
<tr>
<th>Software Process Engineering (SPE)</th>
<th>Business Process Management (BPM)</th>
<th>Workflow Management (WFM)</th>
</tr>
</thead>
</table>
| Relationship with other domains  | - Software Processes are special cases of Business Processes.  
- Software Processes use the Workflow technology to automate the routing of artifacts, the coordination of process activities and roles | - A Business process can be a Software Process since the BPM domain is richer than the SPE.  
- BPM is a superset of the Workflow technology. That latter is used to automate process's repeatable tasks and for artifacts routing | - The Workflow Technology is used by BPM and by SPE to automate process's repeatable tasks and for routing artifacts. |
III - Survey of UML-Based software process modeling languages

UML as a basis for Process Modeling Language

SPESM1.1
SPESM2.0

Framework specialization approaches
Two layers approaches
Combining meta-modeling and executability
UML as a Process Modeling Language (PML)?

- **A Privileged candidate as PML**
- **Understandability**
  - High-Level Constructs and Graphical Representations
  - Multiple perspectives/views
    - Diagrams suitable for process modeling: Class diagrams, Activity and State machine diagrams
- **Expressiveness**
  - Concepts and mechanisms to ensure Proactive & Reactive Control
    - Ensures more than 16/20 control flow patterns [Wohed05],
  - Advanced elements such as loops, conditional branches and exceptions
- **Executability**
  - Concepts with execution semantics (Activity & Action packages)
  - However no universally UML virtual machine adopted!
- **Last but not least: UML is a Standard and a widely adopted formalism**
  - Easier adoption
  - Leveraging existing tools and training supports
Survey on UML-based SPMLs - Evaluation Criteria

- **Expressiveness (Semantic Richness)**
  - Description of basic process elements
    - Activity, Artifact, Role, Human and Tool
  - Activities and steps sequencing
    - Proactive control Vs. Reactive control

- **Understandability**
  - Code like descriptions Vs graphical representations
  - Multiple perspectives

- **Modularization**
  - The ability to combine different chunks of processes
Survey on UML-based SPMLs – Evaluation Criteria

● **Executability**
  - Constructs with operational semantics
  - Support of process execution and simulation

● **Flexibility**
  - To be able to adapt and customize your Software Process to specific projects
    - Adding new kinds of process elements
      - E.g., roles, artifacts, activities, etc.
  - Extending the SPML at lower costs
    - No need to build a new interpreter/compiler for the SPML
  - Humans as deciders for the process workflow
    - Modification of the process model at runtime
SPMLs concerned by the survey

- **Standards**
  - SPEM1.1 [OMG05]
  - SPEM2.0 [OMG07a]

- **Framework specialization approaches**
  - Di Nitto's et al. Approach [Di Nitto02]
  - Promenade [Franch98]

- **Two layers approaches**
  - Chou's Approach [Chou02]

- **Combining meta-modeling and executability**
  - UML4SPM [Bendraou06]
Standards - SPEM1.1

- the OMG's standard for software process modeling
  - Adopted by early 2005

- Comes in form of a MOF 1.3-compliant metamodel that extends concepts from UML1.4 and an UML1.4 Profile

- The building block of the SPEM metamodel is the Process Structure package
  - defines the main structural elements from which a process description may be constructed
Standards - SPEM2.0

- The recently adopted revision of SPEM1.1 [OMG07a]

- SPEM2.0 comes with a new attractive vision
  - Separating all the aspects, contents and materials related to a software development methodology from their possible instantiation in a particular process

- SPEM2.0 comes in the form of MOF-compliant metamodel that reuses UML2.0 Infrastructure and UML2.0 Diagram Interchange specifications.
  - No concept from the UML2.0 Superstructure [OMG 07b] is reused.
    - Was specified as a mandatory requirement in the RFP!!!

- The standard comes also in form of a UML Profile
  - The OCL (Object Constraint Language) rules were not provided with the profile.
Framework specialization approaches

- **Principles**: SPMLs are the result of the specialization of an Object-Oriented framework, with a modeling layer in UML and an implementation layer partially generated from the UML layer.

- **These approaches do neither provide a new metamodel nor extend the UML metamodel**:
  - The UML class diagram is used as a mean to represent process constituents in terms of UML classes.
  - The Activity diagram is used to represent the flow of work.
  - The state machine diagrams (if used) to represent product and activity states.

- **Main proposition**: Di Nitto and Promenade approaches.
Two layers approaches – Chou’s approach

- **Principle:** UML diagrams are used for documenting and modeling the process. A programming language is used to represent these descriptions in a machine readable format
  - No automatic code generation is provided!

- **Main proposition:** the Chou’s approach:
  - The UML class and activity diagrams are used for modeling the process
  - A proprietary programming language is used to support process execution
Combining meta-modeling and executability
- UML4SPM Language

- UML4SPM comes in form of a MOF-compliant metamodel which extends a subset of UML2.0 (Activity and Action packages)
  - A metamodel, a notation and an execution semantics

- Provides the possibility to define executable and simulable software process models
### Summary of the survey of current UML-Based SPMLs

<table>
<thead>
<tr>
<th>Approaches</th>
<th>SPEM1.1</th>
<th>SPEM2.0</th>
<th>Di Nitto's et al. Approach</th>
<th>Promenade</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requirements</td>
<td></td>
<td></td>
<td>Instances of the UML1.3 Class metaclass</td>
<td>Instances of the UML1.3 Class metaclass</td>
</tr>
<tr>
<td>Expressiveness</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Process Elements</td>
<td>Depends on the Compliance point used</td>
<td>Instances of the UML1.3 Class metaclass</td>
<td>Instances of the UML1.3 Class metaclass</td>
<td></td>
</tr>
<tr>
<td>Activity</td>
<td>Work Definition, Activity</td>
<td>Activity / Task Definition</td>
<td>Activity</td>
<td>Task</td>
</tr>
<tr>
<td>Role</td>
<td>Process Role</td>
<td>Role Use / Role Definition</td>
<td>No</td>
<td>Role</td>
</tr>
<tr>
<td>Artifact</td>
<td>WorkProduct</td>
<td>WorkProduct Uses / WorkProduct Definition</td>
<td>Artifact</td>
<td>Document</td>
</tr>
<tr>
<td>Agent</td>
<td>No</td>
<td>No</td>
<td>Human Agent</td>
<td>Agent</td>
</tr>
<tr>
<td>Tool</td>
<td>No</td>
<td>Tool Definition</td>
<td>Software Agent</td>
<td>Tool</td>
</tr>
<tr>
<td>Activitie s/ Actions Coordination</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Proactive Control</td>
<td>start-start, finish-start or finish-finish. The start-finish is lacking</td>
<td>start-start, finish-start or finish-finish and start-finish are lacking</td>
<td>start-start, finish-start or finish-finish. The start-finish is lacking</td>
<td></td>
</tr>
<tr>
<td>Reactive Control</td>
<td>No</td>
<td>No</td>
<td>Use of events</td>
<td>No</td>
</tr>
<tr>
<td>Exception Handling</td>
<td>Not addressed</td>
<td>Not addressed</td>
<td>Not addressed</td>
<td>Not addressed</td>
</tr>
<tr>
<td>Understandability</td>
<td>Good.</td>
<td>Lack. Very complex. One has to deal 3 compliance points, 7 packages and 75 metaclasses.</td>
<td>Lack. No link between the diagrams.</td>
<td>Lack. Process modeler has to combine between class diagrams and precedence graph. No link between the two diagrams</td>
</tr>
</tbody>
</table>
Summary of the survey of current UML-Based SPMLs

<table>
<thead>
<tr>
<th>Approaches</th>
<th>SPEM1.1</th>
<th>SPEM2.0</th>
<th>Di Nitto's et al. Approach</th>
<th>Promenade</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requirements</td>
<td></td>
<td></td>
<td><strong>Lack.</strong> No details on how the code is generated and how different information from the different diagrams are integrated to the Java code. If the state diagram is not complete, code has to be added manually after generation</td>
<td><strong>Lack.</strong> Not addressed</td>
</tr>
<tr>
<td><strong>Executability</strong></td>
<td><strong>Lack.</strong> Out of scope</td>
<td><strong>Lack.</strong> Not addressed.</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Modularization</strong></td>
<td><strong>Lack.</strong> A Unification phase is needed (i.e., renaming of process elements)</td>
<td>Good. Use of Variability mechanisms.</td>
<td><strong>Not addressed</strong></td>
<td><strong>Lack.</strong> Based on renaming classes from different class diagrams</td>
</tr>
<tr>
<td><strong>Metamodel/Profile</strong></td>
<td>A MOF-Compliant metamodel extending UML 1.4 concepts / A UML1.4 Profile is proposed</td>
<td>A MOF-Compliant metamodel extending the UML2.0 Infrastructure / A UML2.0 Profile <strong>Lack.</strong> OCL constraints are not provided</td>
<td><strong>Lack.</strong> No metamodel, no profile provided. Authors provide a predefined UML1.3 class diagram</td>
<td><strong>Lack.</strong> No metamodel, no profile provided. Authors provide a predefined UML1.3 class diagram</td>
</tr>
</tbody>
</table>
# Summary of the survey of current UML-Based SPMLs

<table>
<thead>
<tr>
<th>Approaches</th>
<th>UML4SPM</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Requirements</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Semantic Richness</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Process Elements</strong></td>
<td></td>
</tr>
<tr>
<td>Activity</td>
<td>Software Activity. Many used-defined kinds can be added thanks to &quot;kind&quot; metaclass</td>
</tr>
<tr>
<td>Role</td>
<td>Responsible Role</td>
</tr>
<tr>
<td>Artifact</td>
<td>WorkProduct</td>
</tr>
<tr>
<td>Agent</td>
<td>Agent and Team</td>
</tr>
<tr>
<td>Tool</td>
<td>Tool</td>
</tr>
<tr>
<td><strong>Activities/Actions Coordination</strong></td>
<td></td>
</tr>
<tr>
<td>Proactive Control</td>
<td>Ensured thanks to the combination of control flow, object flow, Invocation Actions and control nodes. Kinds of precedence ensured: start-start, finish-start or finish-finish and start-finish.</td>
</tr>
<tr>
<td>Reactive Control</td>
<td>Ensured through events (Message, Change and Time events) and actions (AcceptEvent and SendSignal actions)</td>
</tr>
<tr>
<td>Exception Handling</td>
<td>Use of RaiseExceptionAction and ExceptionHandler concepts</td>
</tr>
<tr>
<td>Advanced Constructs (Tool Invocations, Agent Communication, Loops, Conditionals)</td>
<td>Loops, Conditional, Decision Nodes, DataStoreNode (WorkProduct persistency and versioning)</td>
</tr>
</tbody>
</table>
## Summary of the survey of Current UML-Based SPMLs

<table>
<thead>
<tr>
<th>Approaches</th>
<th>UML4SPM</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Requirements</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Understandability</strong></td>
<td>Good: Simple metamodel with a minimal set of concepts, reuse of UML2.0 diagrams with a minor customization in order to represent features proper to software process modeling. Requires a minimum knowledge of UML2.0</td>
</tr>
<tr>
<td><strong>Precision</strong></td>
<td>Good: Two levels of abstractions. Process view for a global view of process's activities and Activity view for a detailed description of the activity in terms of actions. The unit of work is the Software Activity (can be specialized using the &quot;Kind&quot; metaclass i.e., a process, a phase, iteration, etc.). Action is the atomic unit of work in UML4SPM.</td>
</tr>
<tr>
<td><strong>Executability</strong></td>
<td>Good: Two approaches proposed. The first, transformation of UML4SPM process models towards BPEL for execution. The second, direct process model execution using UML4SPM process engine. Lack: Need for configuration phase in the BPEL approach. Lack of resource management support with the current UML4SPM process engine</td>
</tr>
<tr>
<td><strong>Modularization</strong></td>
<td>Through the use of CallBehaviorAction in order to compose between different activities. More configuration in case of complete process models.</td>
</tr>
<tr>
<td><strong>Formality</strong></td>
<td>Translational semantics to Petri-nets approach possible</td>
</tr>
<tr>
<td><strong>Metamodel/ Profile</strong></td>
<td>A MOF-Compliant metamodel extending UML2.0 Superstructure concepts. No profile is proposed.</td>
</tr>
</tbody>
</table>
Summary

- **SPM1.1**
  - Expressiveness: 10
  - Formality: 5
  - Understandability: 1
  - Modularization: 1
  - Executive: 1

- **SPM2.0**
  - Expressiveness: 10
  - Formality: 5
  - Understandability: 1
  - Modularization: 1
  - Executive: 1

- **Di Nitro's Approach**
  - Expressiveness: 10
  - Formality: 5
  - Understandability: 1
  - Modularization: 1
  - Executive: 1

- **Promenade**
  - Expressiveness: 10
  - Formality: 5
  - Understandability: 1
  - Modularization: 1
  - Executive: 1

- **Chou's Approach**
  - Expressiveness: 10
  - Formality: 5
  - Understandability: 1
  - Modularization: 1
  - Executive: 1

- **UML4SPM**
  - Expressiveness: 10
  - Formality: 5
  - Understandability: 1
  - Modularization: 1
  - Executive: 1
References (1)

- [Bendraou07a], UML4SPM : un langagde de modélisation de procédés de développement logiciel exécutable et orienté modèle, PhD Thesis of University Paris 6, 6 septembre 2007 (in english)
- [Chou02], S.-C. Chou, A process modeling language consisting of high level UML diagrams and low level process language, Journal of Object Technology 1, 2002, 4, pp. 137-163
- [Di Nitto02], Di Nitto E. et at. "Deriving executable process descriptions from UML", in Proc. of the 24th Inter. Conf. on Software Engineering (ICSE’02), Orlando, Florida 2002, ACM Press
- [Humphrey89], W.S. Humphrey, "The Software Engineering Process: Definition and Scope", in Proceedings of the 4th International Software Process Workshop on Representing and Enacting the Software Process, Devon, United Kingdom, 1989
References (2)

- [Wohed05], Wohed, P., van der Aalst, W., Dumas, M., ter Hofstede, A. & Russell, N., "Pattern-based analysis of UML activity diagrams", in Proceedings of the 25th International Conference on Conceptual Modeling (ER'2005), Springer, Klagenfurt, Austria