Performance Management Group

fortiss

Munich, Germany, 2015-10-06

A Performance Model Management Repository Based on the Palladio Component Model

Symposium on Software Performance 2015

Alexandru Danciu [1], Andreas Brunnert[1], Helmut Krcmar[2]

[1]fortiss GmbH, [2]Technical University of Munich (TUM)

Agenda

- Motivation & Vision
- Extensions to PCM
  - Versioning of Components and Interfaces
  - Central Repository for Components
  - Handling Resource Demands
- Outlook

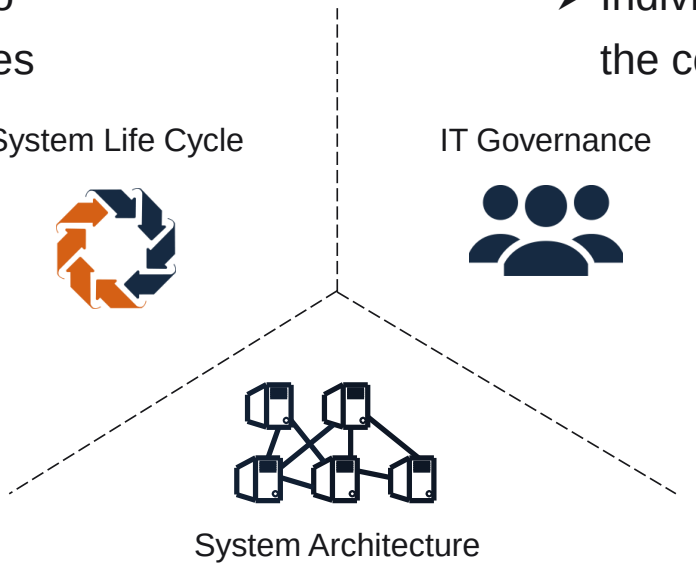
Agenda

- Motivation & Vision
- Extensions to PCM
  - Versioning of Components and Interfaces
  - Central Repository for Components
  - Handling Resource Demands
- Outlook

Motivation & Vision

Challenge for applying performance models in industrial practice is the organizational complexity

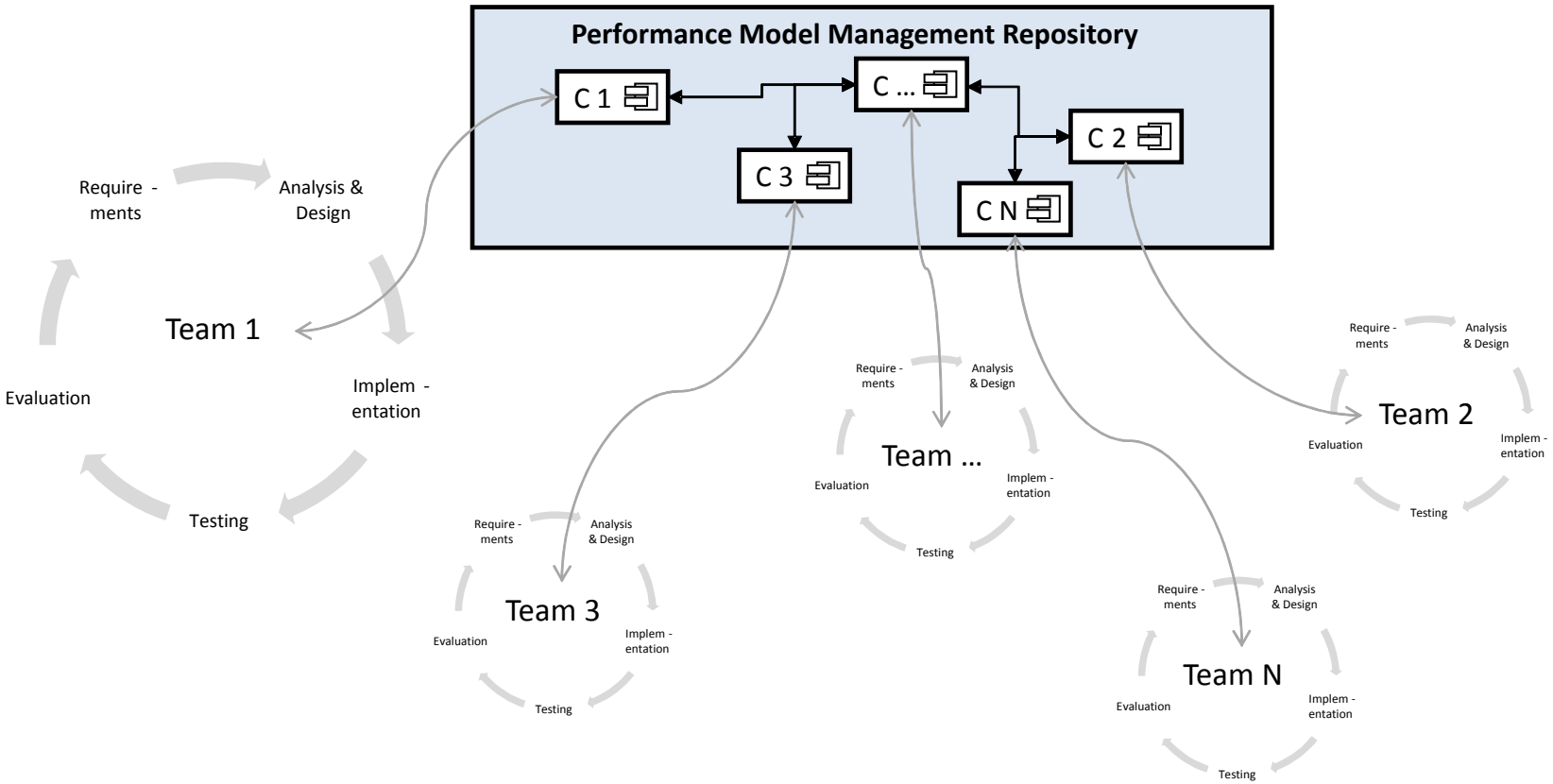
- Components adhere to diverging release cycles
- Individual components under the control of different teams



- Increased number of components and dependencies

Motivation & Vision

- To introduce an integration server for performance models to support the collaboration of distributed teams within an organization



Agenda

- Motivation & Vision
- Extensions to PCM
  - Versioning of Components and Interfaces
  - Central Repository for Components
  - Handling Resource Demands
- Outlook

Versioning of Components and Interfaces

Types of Versions

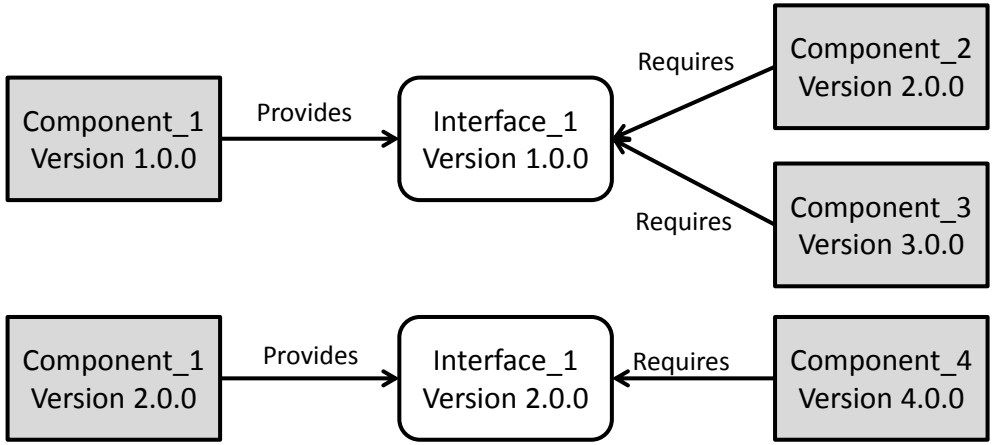
- Within software configuration management (SCM) the term version is used on different abstraction levels:
  - **Implicit versions:**
    - Changes to software artifacts or model elements recorded by version control systems
    - Automatically generated
  - **Explicit versions:**
    - Defined as part of a release management
    - Specified by users

- Different types of verisons with regard to the nature of a change:
  - **Interface versioning:** changes in the interaction of components
  - **Implementation versioning:** changes in the source code of a component

Versioning of Components and Interfaces

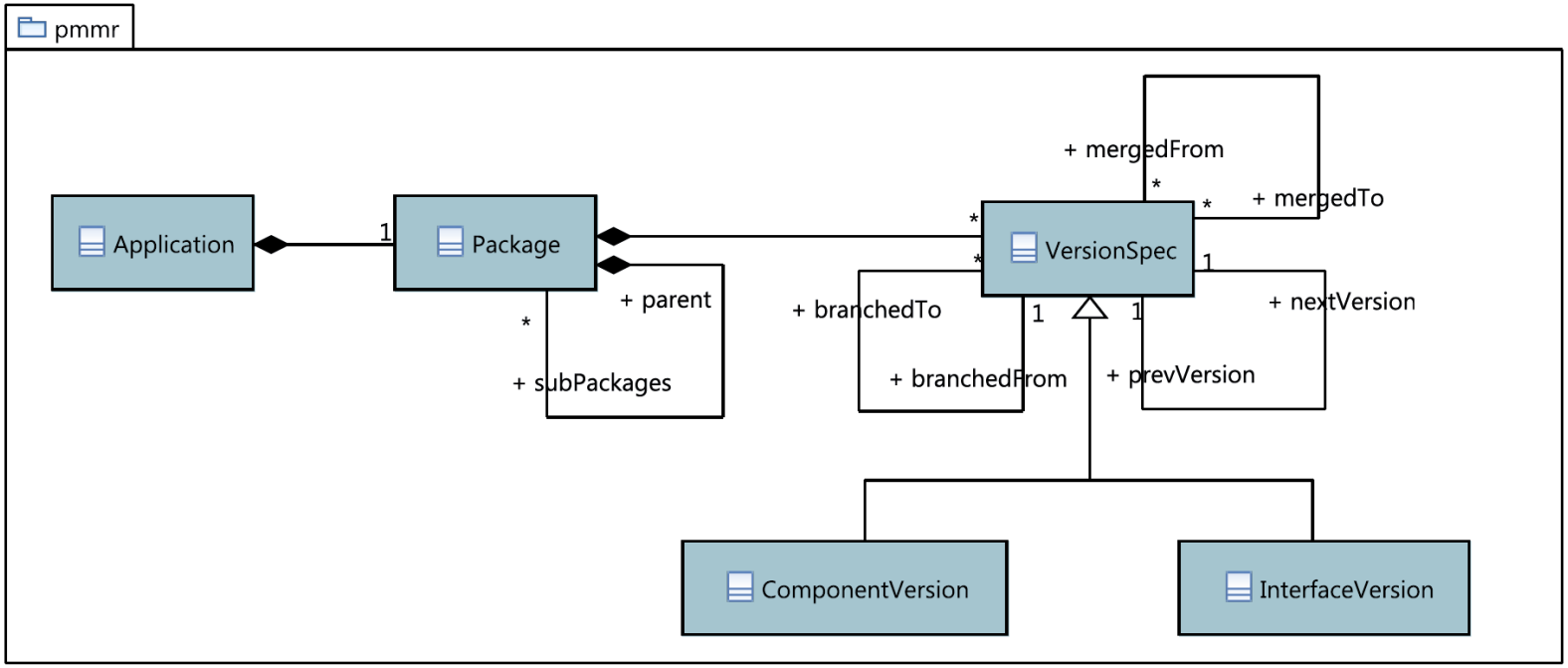
Versioning within a PMMR

- A PMMR requires the existence of **explicit versions** to support specifying and maintaining different releases of **components and interfaces** simultaneously



PCM Meta-Model Extensions for Managing Component and Interface Versions

- Contents of a PMMR are subordinated to an *Application* element
- *Version Specications* are organized in a hierarchical structure of *Package* elements



Agenda

- Motivation & Vision
- Extensions to PCM
  - Versioning of Components and Interfaces
  - Central Repository for Components
  - Handling Resource Demands
- Outlook

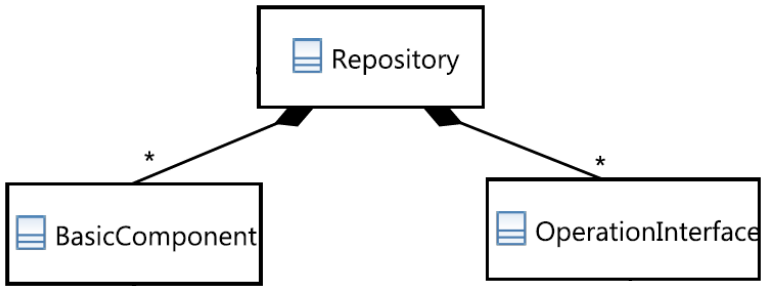
Central Repository for Components

Conditions Imposed by PCM

- Components and interfaces are contained in a *Repository*
  - Components are represented in PCM as *BasicComponent* elements
  - Interfaces are represented in PCM as *OperationInterface* elements

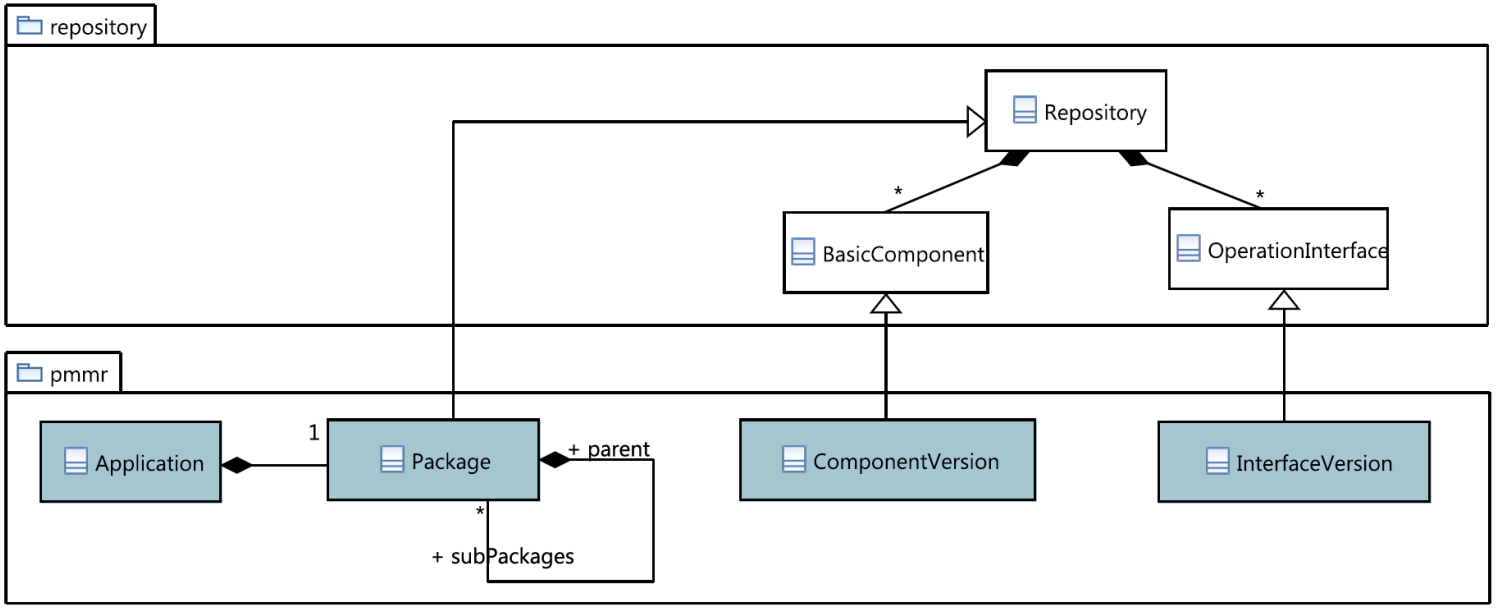
➢ These elements cannot be contained by any other class in an EMF model

- However, standard editors generated by EMF only support creating and displaying these elements when a containment relationship exists



PCM Meta-Model Extensions for Managing Components and Interfaces in a PMMR

- Components and interface implementations can be organized in a hierarchical package structure avoiding storing them in one large collection



Central Repository for Components

Interaction with the PMMR

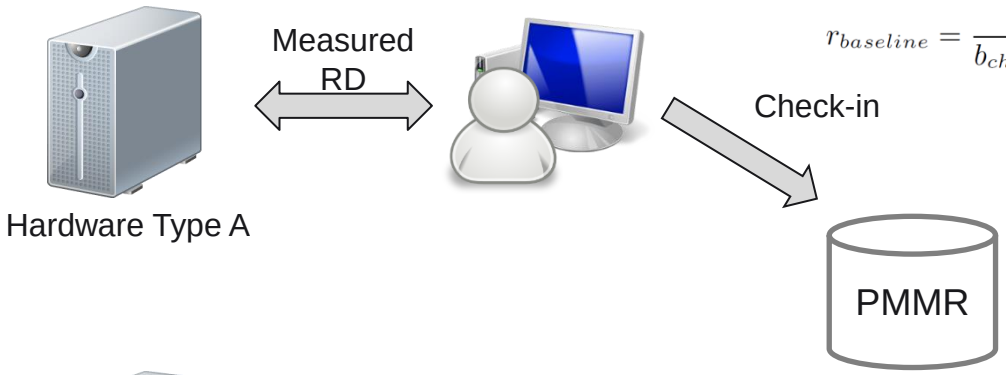
- PMMR contents persisted in EMFStore and organized in projects
- Palladio-Bench extended to support connecting to EMFStore for checking out local copies
- User management for restricting the access to the PMMR provided
- Changes to contents of local projects are tracked and committed to the repository server
- To support adding a component instance to multiple repository models object clones are created and kept synchronized with the PMMR

Agenda

- Motivation & Vision
- Extensions to PCM
  - Versioning of Components and Interfaces
  - Central Repository for Components
  - Handling Resource Demands
- Outlook

Handling Resource Demands

- Components managed by a PMMR can specify hardware-specific resource demands
- Resource demands stored in a PMMR are specified relative to a common baseline

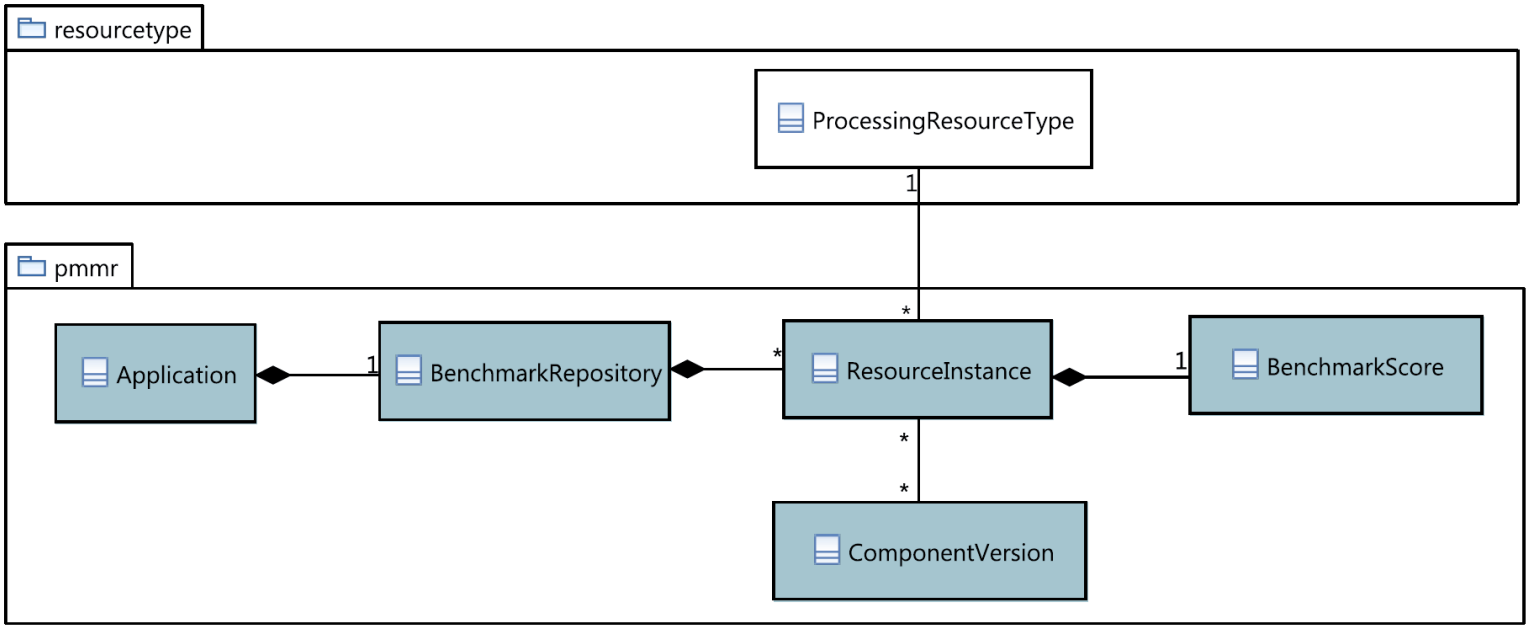


$$r_{baseline} = \frac{b_{baseline}}{b_{checkinbenchmarkvalue}} * r_{checkinvalue}$$

$$r_{checkoutvalue} = \frac{b_{checkoutbenchmarkvalue}}{b_{baseline}} * r_{baseline}$$

PCM Meta-Model Extensions for Managing Rresource Demands

- Palladio-Bench provides a predefined set of *ProcessingResourceType* objects
- *ResourceInstance* elements represent specific hardware resources, of a certain type (such as CPU), manufactured by a specific hardware vendor



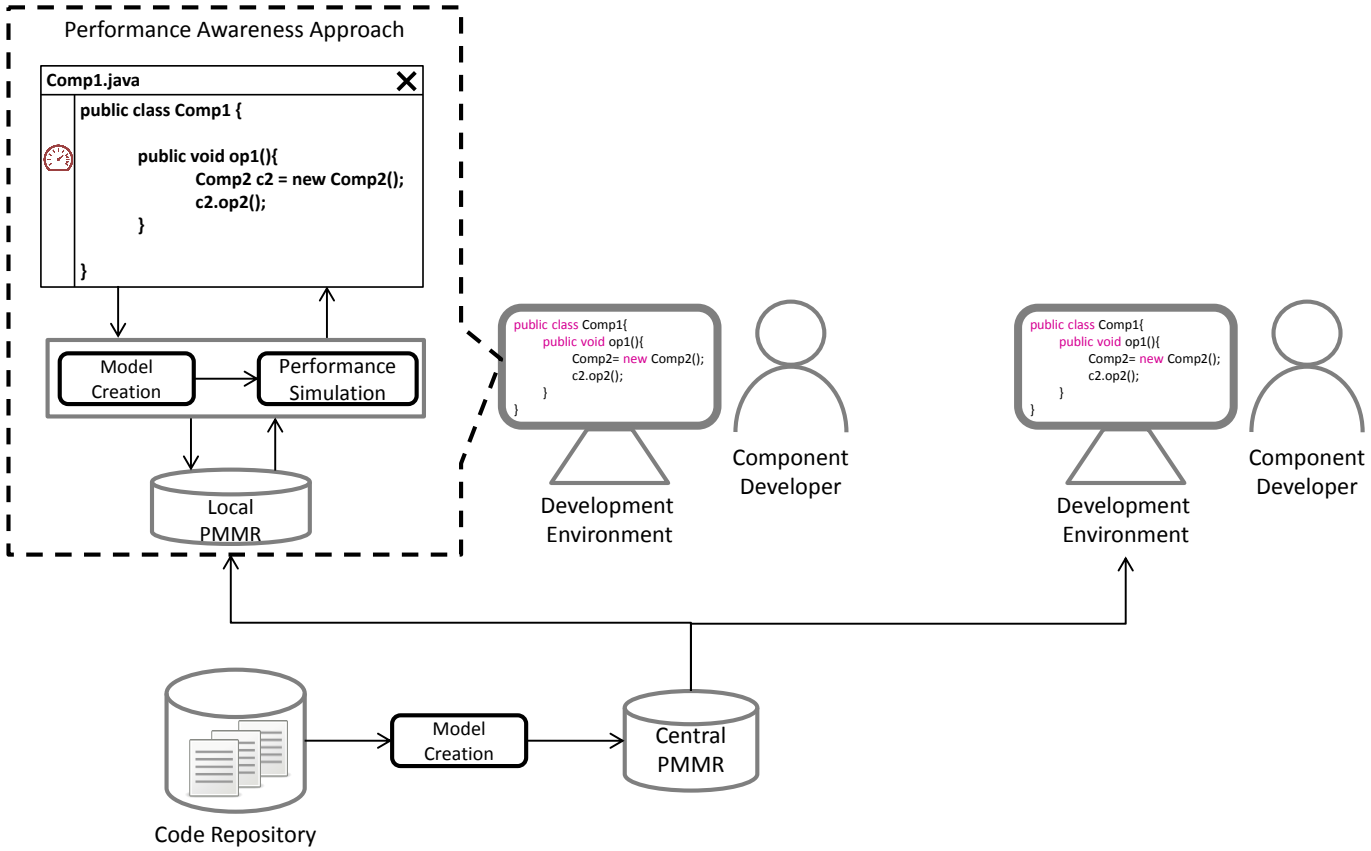
Agenda

- Motivation & Vision
- Extensions to PCM
  - Versioning of Components and Interfaces
  - Central Repository for Components
  - Handling Resource Demands
- Outlook

Outlook

Evaluation

- Performance awareness as use case for evaluating the PMMR



CONTACT US

Alexandru Danciu

danciu@fortiss.org

performancegroup@fortiss.org
pmw.fortiss.org