Developing and Evolving a Multi-Agent System Product Line: An Exploratory Study

Ingrid Nunes[1], Camila Nunes[1], Uirá Kulesza[2], Carlos Lucena[1]

[1]Pontifical Catholic University of Rio de Janeiro (PUC-Rio)

[2]New University of Lisbon & CESAR

Purpose

- Multi-agent System Product Line (MAS-PL) development
  - Evolving system seen as a SPL
  - Different versions of a Web Application
    - Each version has new features, including agency features
- Discussion of relevant questions
  - Modularization of agency features
  - Incorporation of autonomous behavior in a traditional web-based system
  - Which SPL models are useful and essential to specify and document a MAS-PL

Outline

- ExpertCommittee MAS-PL
- Lessons Learned
- Related Work
- Conclusions
- Future Work

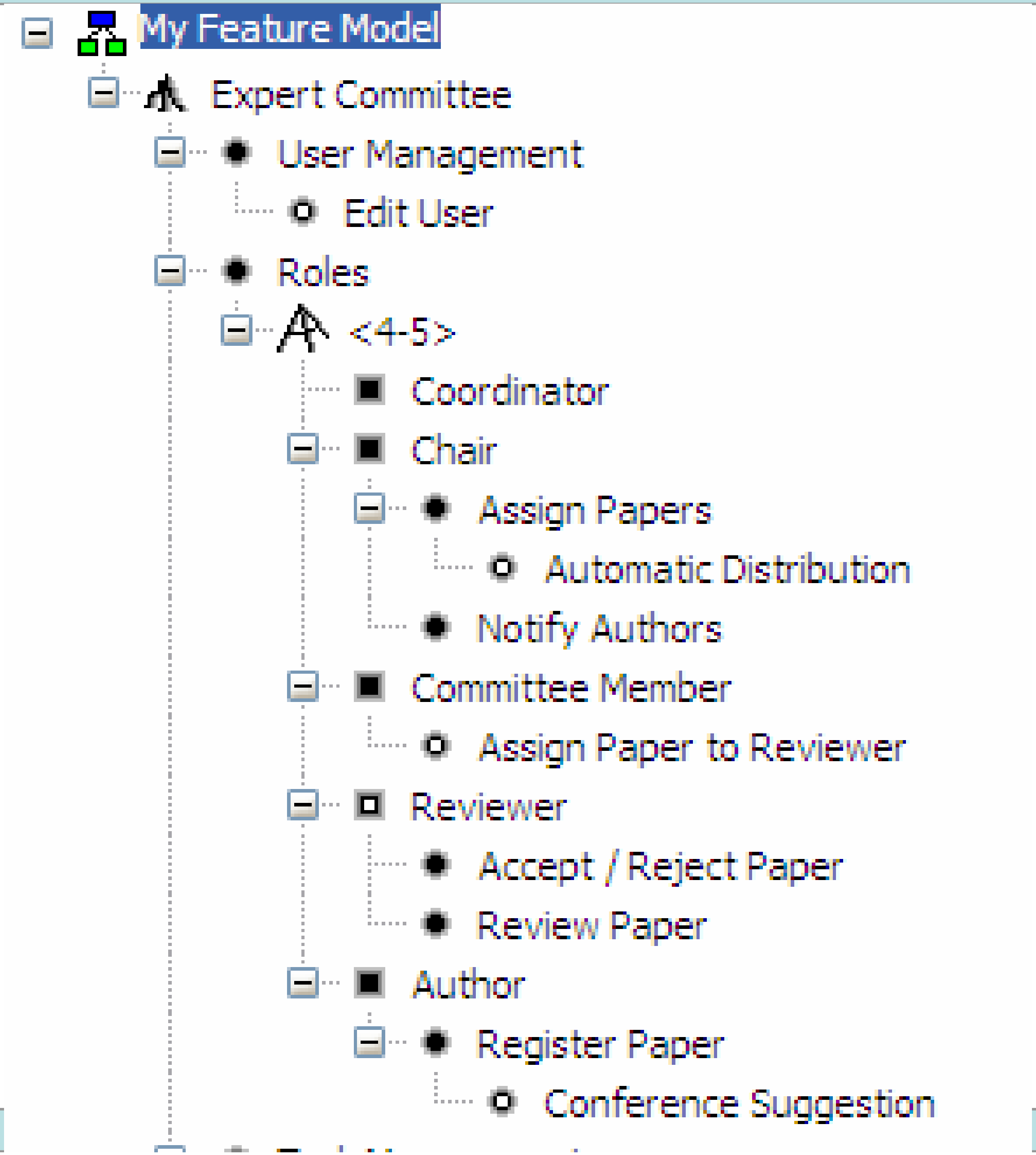
ExpertCommittee (EC)

- Conference Management System
- Typical web-based application
- Functionalities to support the complete process of the management of conferences and workshops
  - Paper submission
  - Reviewing processes
- Examples
  - EasyChair (http://www.easychair.org/)
  - JEMS (https://submissoes.sbc.org.br/)

EC Versions

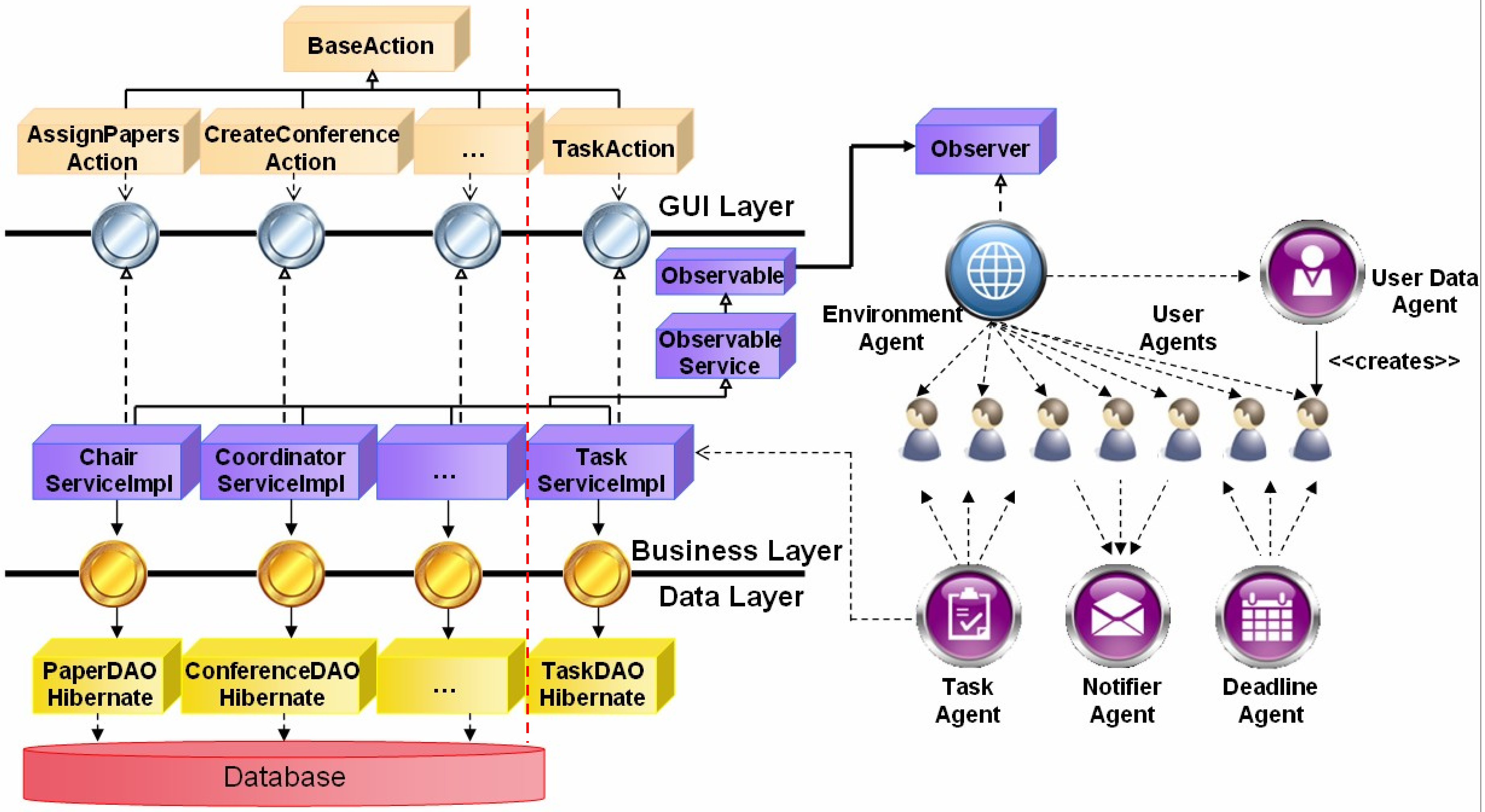
Version	Description
Version 1	Typical web-based application Represents our MAS-PL core Mandatory features that support the conference management process
Version 2	Additions: - reviewer role and the functionalities related to it - edit user and assign review to reviewer - autonomous behavior (agents) features
Version 3	Refactoring of Version 2 Improvement of the modularization of some agency features in order to make possible the automatic product derivation

Feature Model



- My Feature Model
  - Expert Committee
    - ● User Management
      - ○ Edit User
    - ● Roles
      - <4-5>
        - ■ Coordinator
        - ■ Chair
          - ● Assign Papers
            - ○ Automatic Distribution
          - ● Notify Authors
        - ■ Committee Member
          - ○ Assign Paper to Reviewer
        - □ Reviewer
          - ● Accept / Reject Paper
          - ● Review Paper
        - ■ Author
          - ● Register Paper
            - ○ Conference Suggestion

MAS-PL Architecture



Patterns

- Layer architectural pattern
  - GUI – process web requests
  - Business – structure and organize business services
  - Data – database access classes
    - DAO design pattern
- Observer pattern
  - Services are observable objects that notify observers about their actions
  - Environment Agent receives notifications and broadcast to the user agents
  - Allow the (un)plugability of the agency features
- Role pattern
  - Models context-specific views of an object as separate role objects which are dynamically attached to and removed from the core object

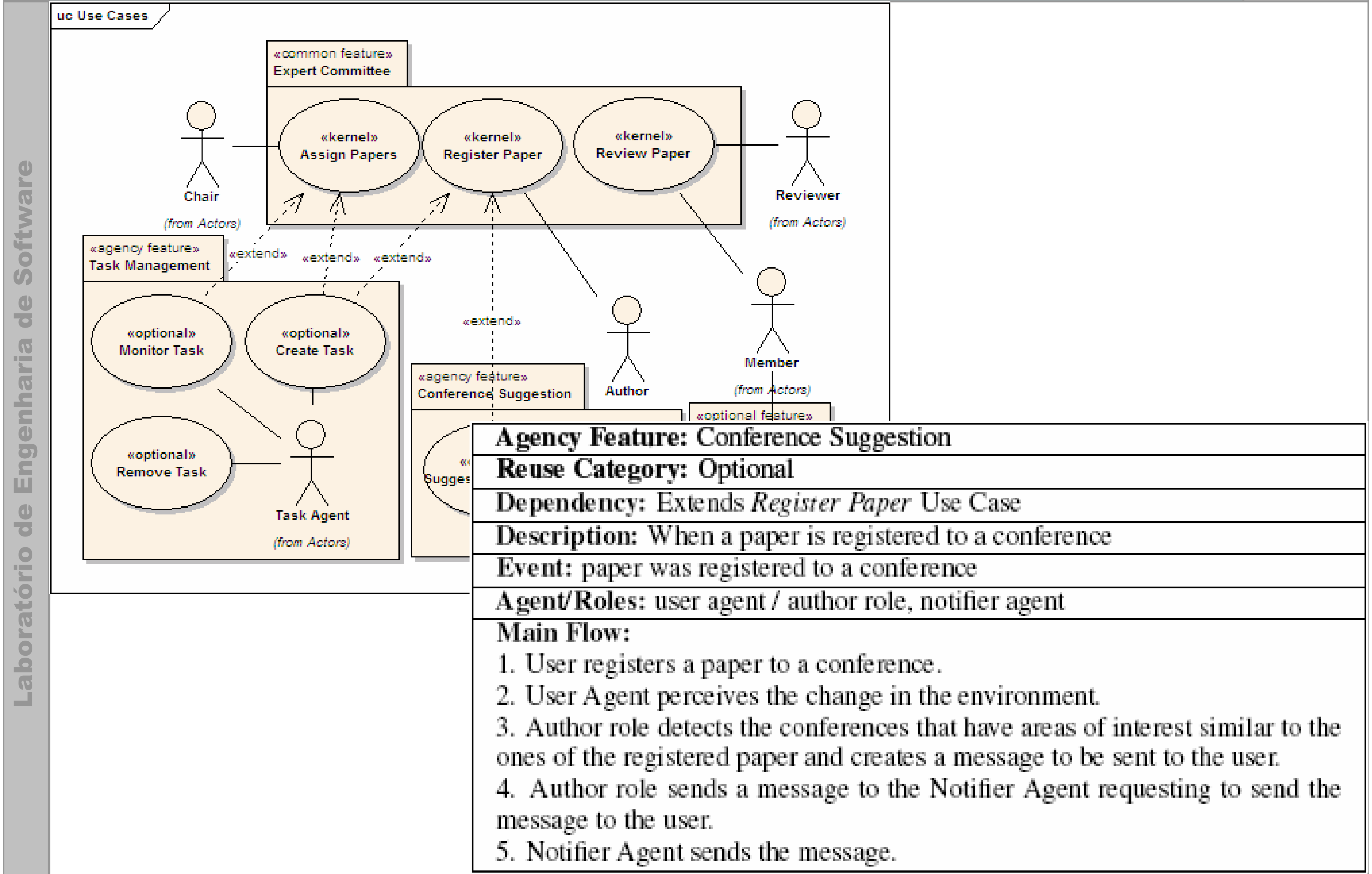
Lessons Learned

- MAS-PL Features Types
  - New conference management features
    - Typical features addressed in SPL
  - New autonomous behavior
    - Introduction of agents
  - New behaviors and roles for an agent
    - Features that have impact inside of agents

Lessons Learned

- Many SPL methodologies have been proposed
  - Too abstract (lack of design details) OR
  - Concepts and techniques from object-oriented and component-based paradigm
  - Do not detail or barely detail the modeling and documentation
    - Agents features
    - Role features

Lessons Learned

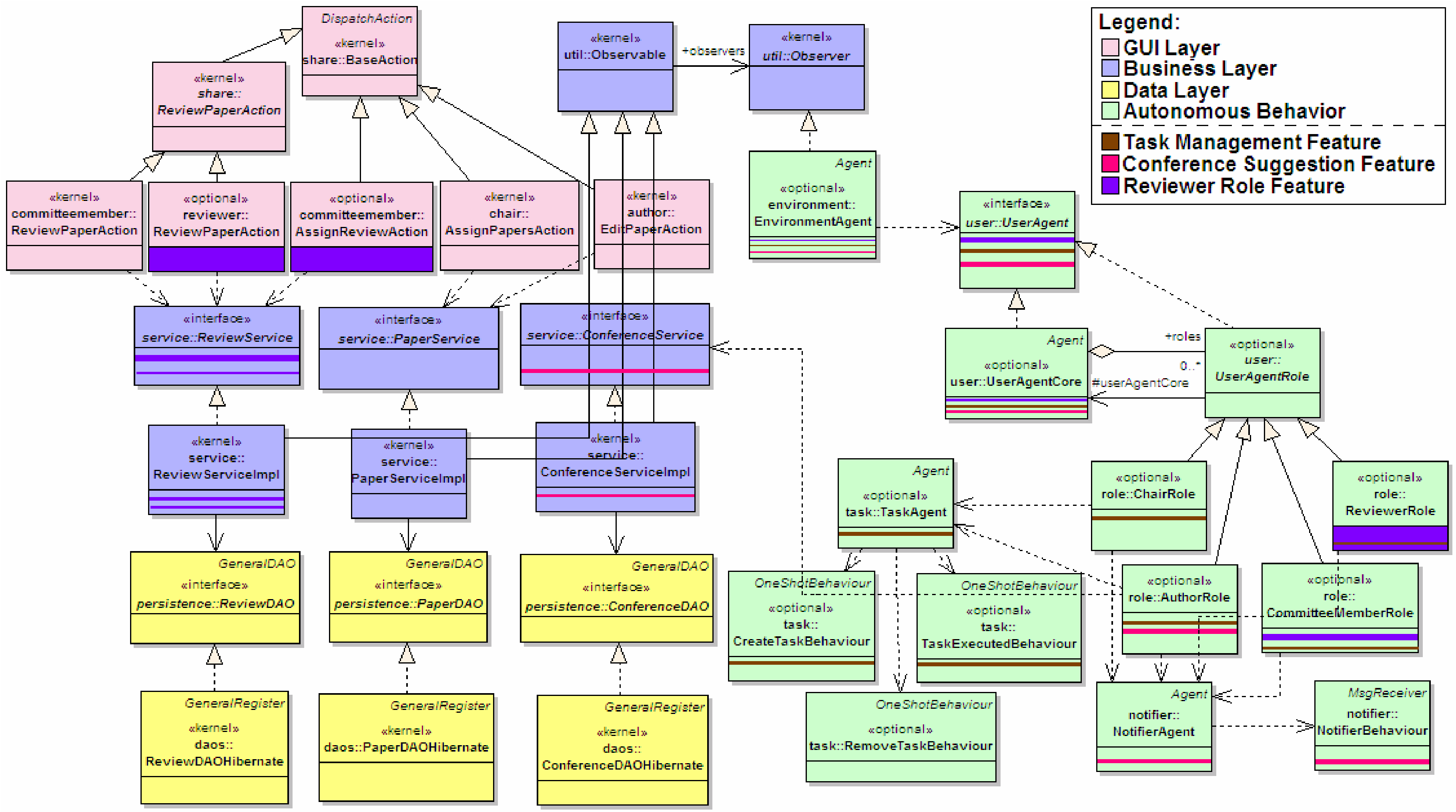


Agency Feature:	Conference Suggestion
Reuse Category:	Optional
Dependency:	Extends *Register Paper* Use Case
Description:	When a paper is registered to a conference
Event:	paper was registered to a conference
Agent/Roles:	user agent / author role, notifier agent
Main Flow:	1. User registers a paper to a conference. 2. User Agent perceives the change in the environment. 3. Author role detects the conferences that have areas of interest similar to the ones of the registered paper and creates a message to be sent to the user. 4. Author role sends a message to the Notifier Agent requesting to send the message to the user. 5. Notifier Agent sends the message.

Lessons Learned

- Aspect-oriented Refactoring
  - Problem: Crosscutting Concerns
    - Tangled code
    - Spread Code
    - Replicated Code
  - Difficult to maintain and to reuse
- Interesting situations to adopt AOP techniques
  - modularization of the glue-code that integrates
    - web-based system (base code)
    - agent features (new variable agency features)
  - modularization of the agent roles

Lessons Learned



Legend:

- ■ GUI Layer
- 🟦 Business Layer
- 🟨 Data Layer
- 🟩 Autonomous Behavior
- 🟫 Task Management Feature
- 🟥 Conference Suggestion Feature
- 🟪 Reviewer Role Feature

Related Work

- Dehlinger & Lutz
  - A Product-Line Requirements Approach to Safe Reuse in Multi-Agent Systems
  - Extensible agent-oriented requirements specification template for distributed systems that supports safe reuse
    - Role Schema
    - Role Variation Point Schema
- Peña et al
  - Designing and managing evolving systems using a MAS product line approach
  - View different instances of a system as it evolves as different “products” in a SPL
  - Based on an extension of MaCMAS

Conclusions

- Exploratory study
  - Development and evolution of a MAS-PL
  - Traditional web-based system with new agency features
- Feature model – drives incorporation of new features
- User agents and roles – features modularization
- Architecture that integrates agency features and traditional web-based system
- Lessons learned
  - Feature types
  - Improve the modularization of agency features with AO techniques
  - Adaptation of existing SPL methodologies

Future Work

- Propose a MAS-PL development methodology
  - Extensions of SPL methodologies
  - Usage of MAS techniques
- MAS-PL Empirical Study
  - Quantitatively and qualitatively comparison of OO and OO+AO implementations using software metrics
- Development of new MAS-PL case studies of web-based systems

Questions?

Ingrid Nunes
ioliveira@inf.puc-rio.br
Camila Nunes
cnunes@inf.puc-rio.br
Uirá Kulesza
uira@cesar.org.br
Carlos Lucena
lucena@inf.puc-rio.br