Interaction platform administration strategies: Practice and experience
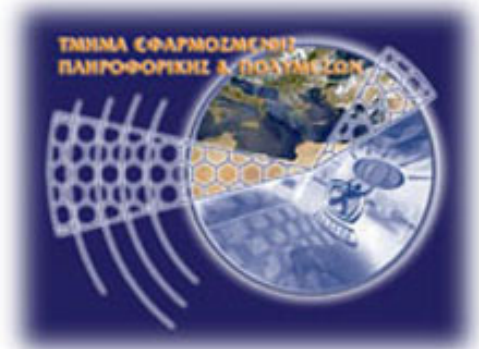
Akoumianakis D., Milolidakis G., Kotsalis D., Vellis G.

Dept. Applied Information Technology & Multimedia Technological Education Institution of Crete, Estavromenos 715 00 Heraklion – Crete, GREECE

Overview

- The issue under investigation
  - Managing diverse collections of interaction elements
    - Toolkits
    - Custom-widgets
  - Interoperation models / APIs
    - Fusing diverse collections

- Proposed solution
  - Platform administration strategies
    - Augmentation
    - Expansion
    - Integration
    - Abstraction

- Context – eKoNEΣ
  - R&D project co-funded by GSRT (1.2 – R&D consortia in strategic areas of national priority)
  - An 'electronic village' of local interest
    - Thematic focus on tourism

Standard GUI toolkits

- Standardized set of implemented widgets
    - Widgets = presentation + behavior + dialogue control
    - Basic widgets can be used to build composite objects
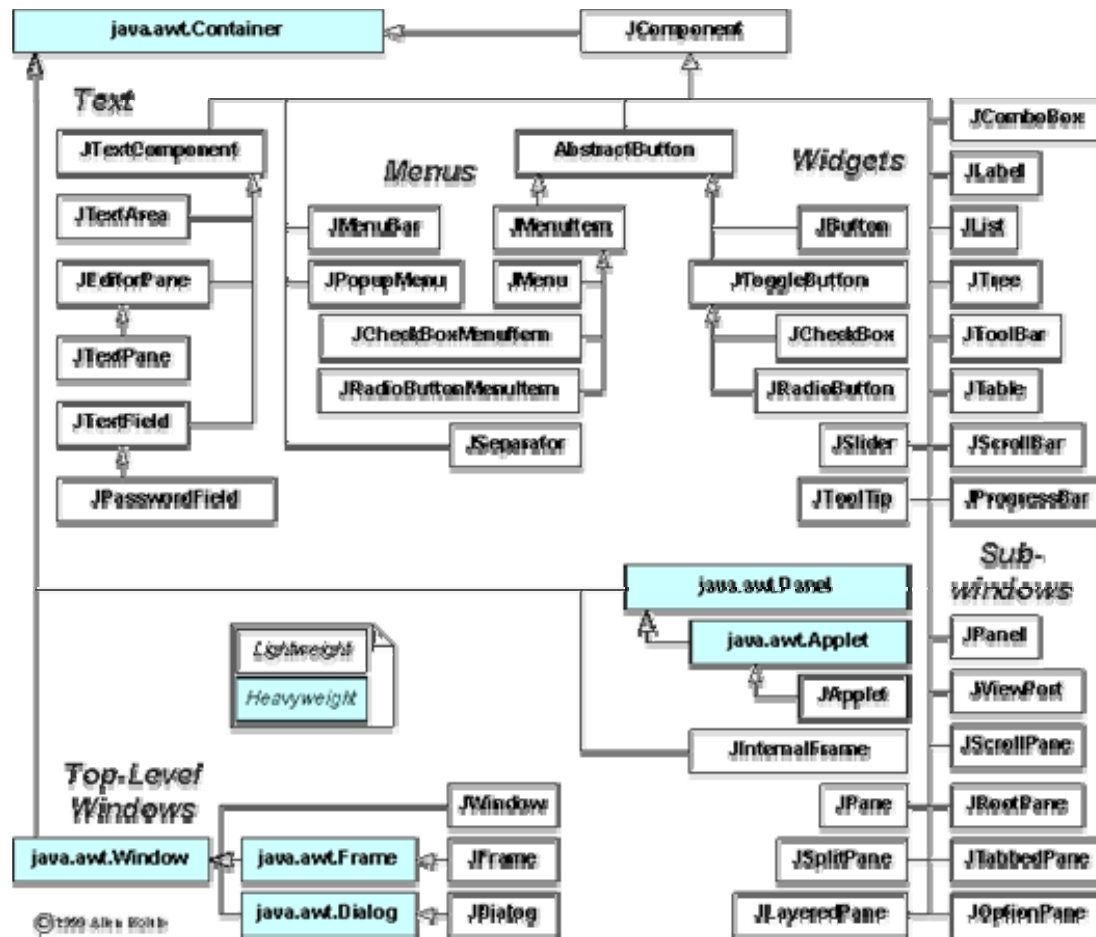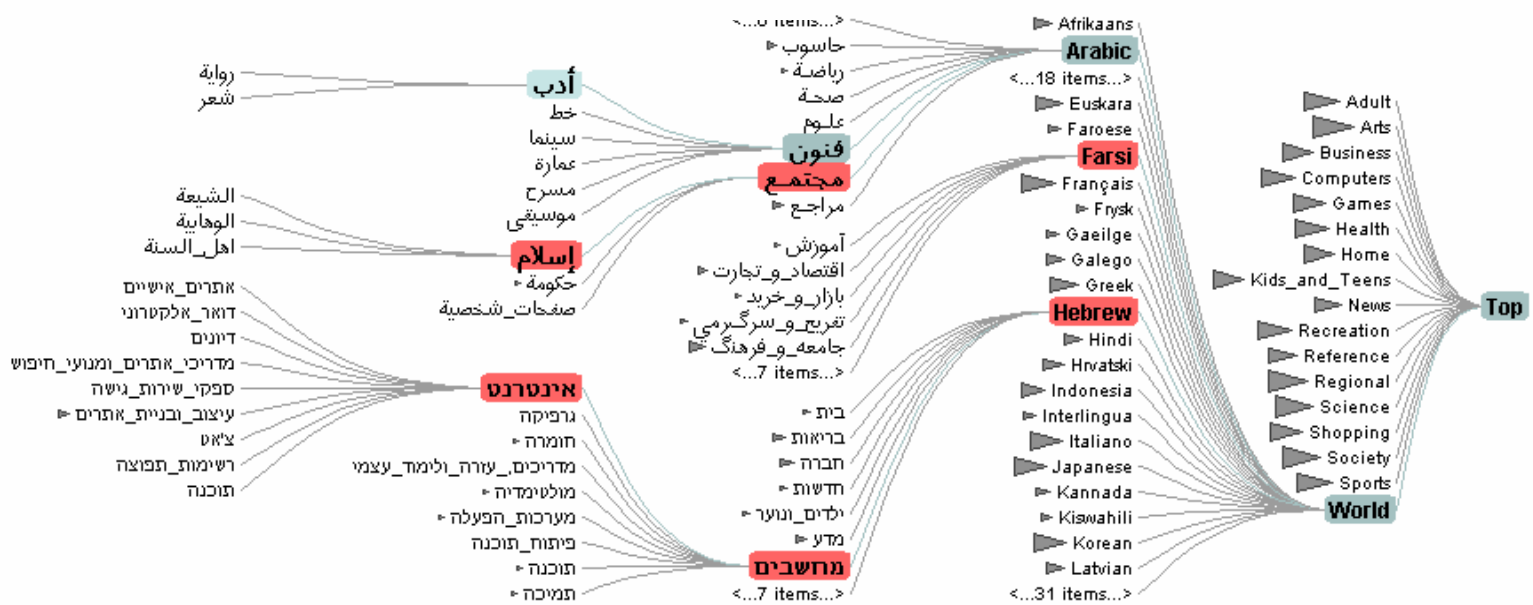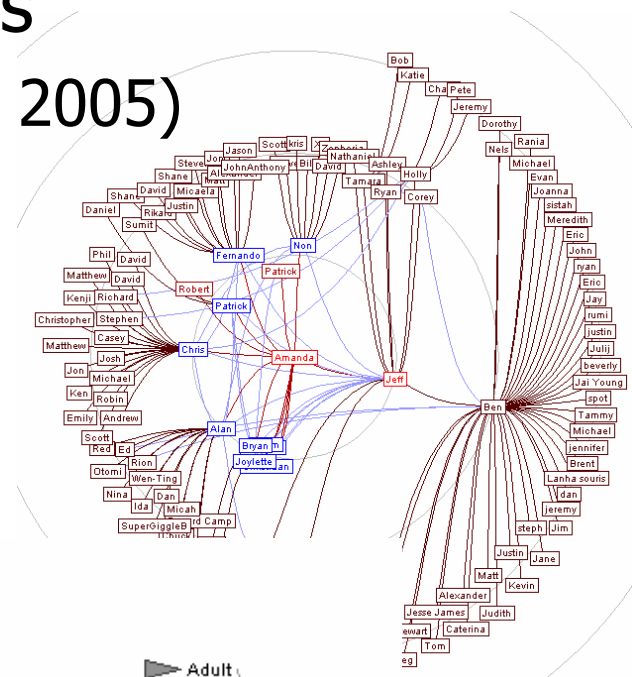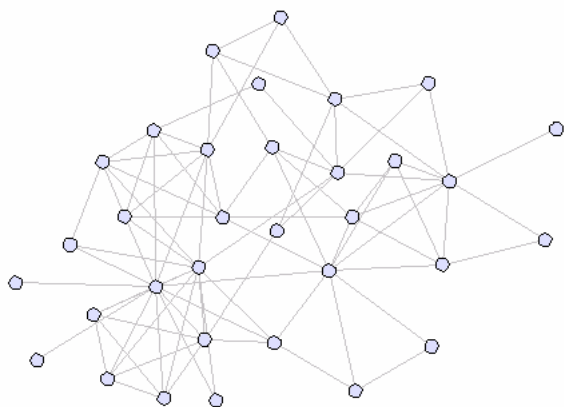
Toolkits (Cont.)

- Advanced 'cutting edge' toolkits
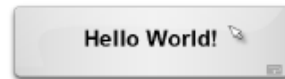 - Prefuse (Heer, Card and Landay, 2005)

Custom widgets

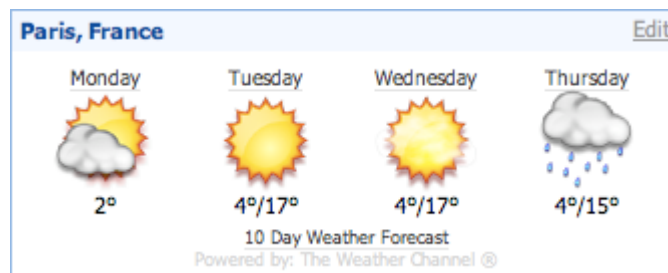- Defined to facilitate specific interactive 'look & feel' or behaviors

  - Buttons

  - To-do lists

  - The weather forecast widget

# *Custom widgets (Cont.)*

- Some novel widgets

Domain-specific components

- Frequently, a design will require the implementation of a domain-specific interaction component (e.g. to implement a domain-specific metaphor)

- Implications
  - Not easy (in case of serious efforts)
    - Typically, it requires
      - Manipulation of abstract and top-level container objects
      - Definition of suitable layout manager (containment strategy)
      - Use of non-native components
  - Not portable across platforms

- What happens if the primitive objects are not supported by one single library?
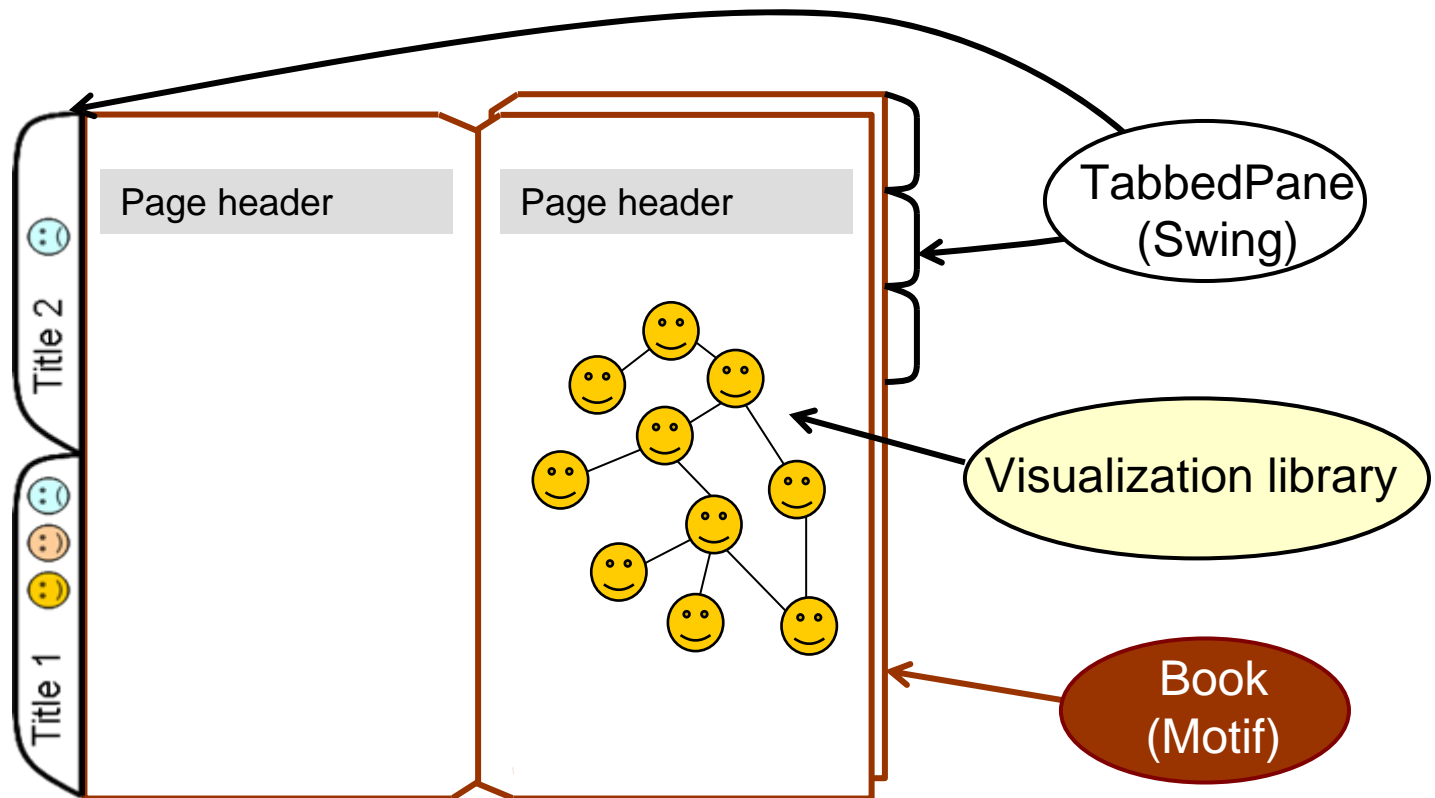
Example

- Composing a 'book' metaphor using existing interaction components available through different toolkits
  - Book
  - Multi
  - Visua

Page header

Page header

Title 2

Title 1

Messages   Community
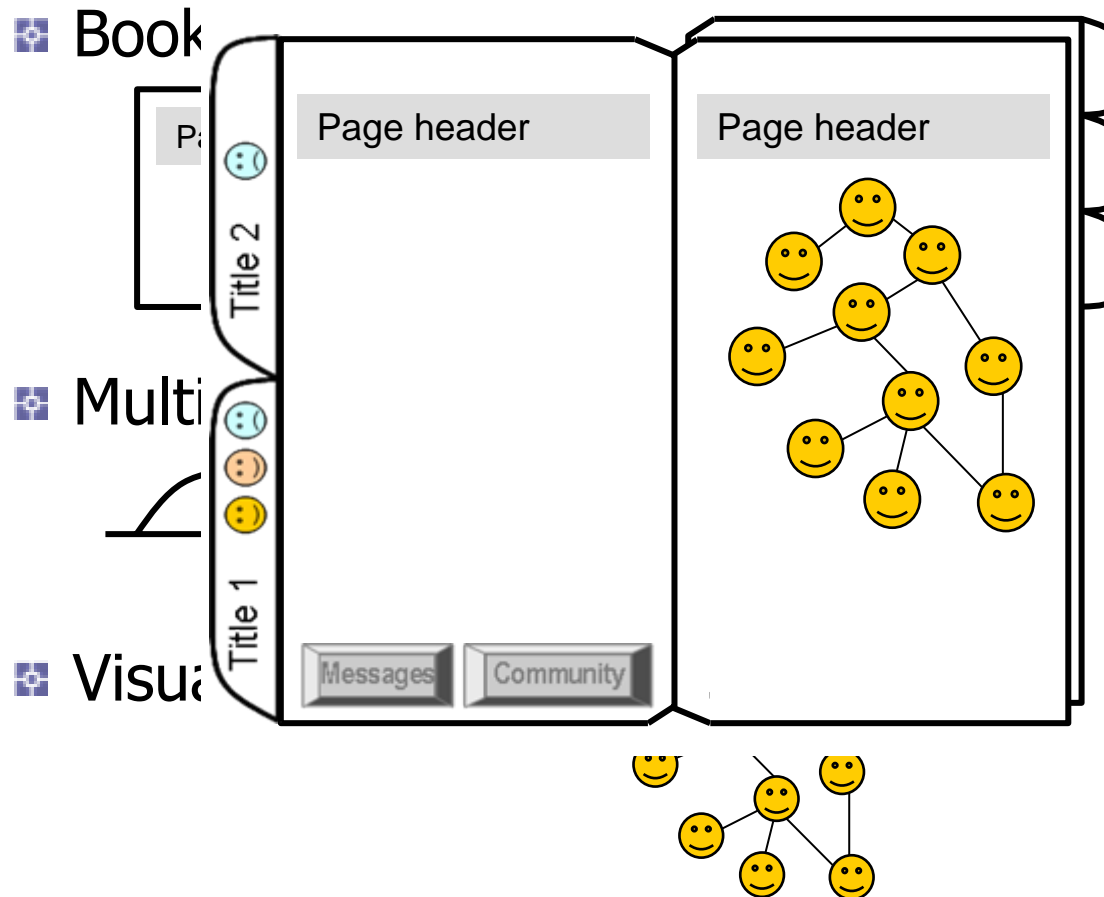
Some issues

- Given the diverse collections of interaction components, how can developers systematize the way the make use of them

  - In other words – can we build mixed-toolkit hierarchies

Page header

Page header

Title 2

Title 1

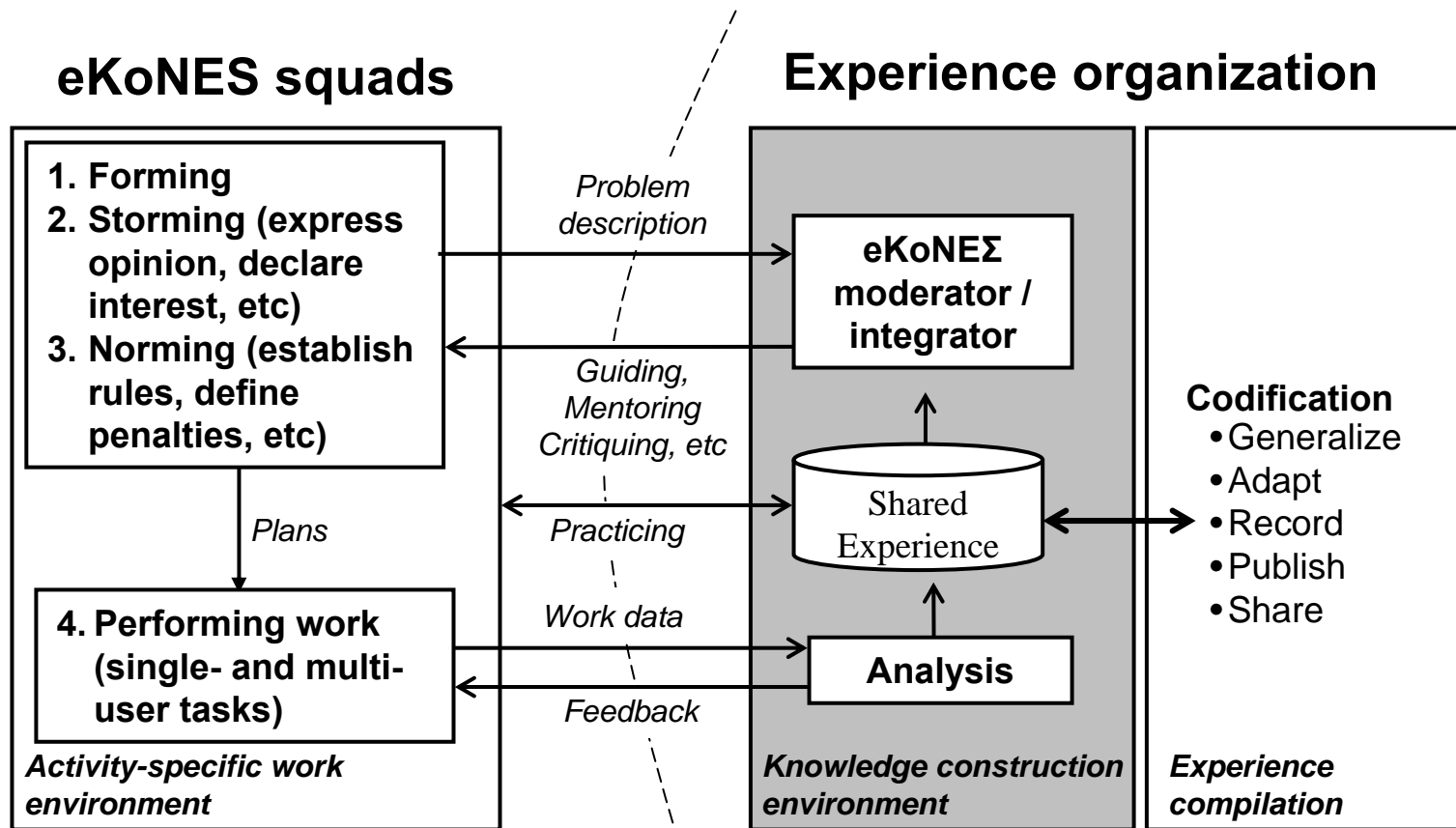TabbedPane
(Swing)

Visualization library

Book
(Motif)

Context of our work

- eKoNEΣ – an electronic village of local interest on tourism
  - A work-oriented 'experience factory' supporting social interaction
    - New product / package composition by negotiating, coordinating and finally integrating shared resources
    - Visualizing collaborative performance of virtual squads by evolution phase
    - Synchronous collaborative sessions to address conflicts
- Scenario
  - End users populate requests, articulate demand and 'villagers' join forces as virtual squads to fulfill the demand
    - Incremental compilation of new 'added value' packages owned by the squad

# *The 'social' experience factory*

- Architectural underpinnings

eKoNES squads

Experience organization

1. **Forming**	
2. **Storming (express opinion, declare interest, etc)**	
3. **Norming (establish rules, define penalties, etc)**	

Problem description

eKoNEΣ moderator / integrator

Guiding, Mentoring Critiquing, etc

Plans

4. **Performing work (single- and multi-user tasks)**

Practicing

Work data

Feedback

Shared Experience

Analysis

Codification
- Generalize
- Adapt
- Record
- Publish
- Share

Activity-specific work environment

Knowledge construction environment

Experience compilation

Platform administration

- Platform administration process
  - A specific workflow = what is to be done
  - 'Platform administrator' role = who is doing it
  - Artifacts = what is produced
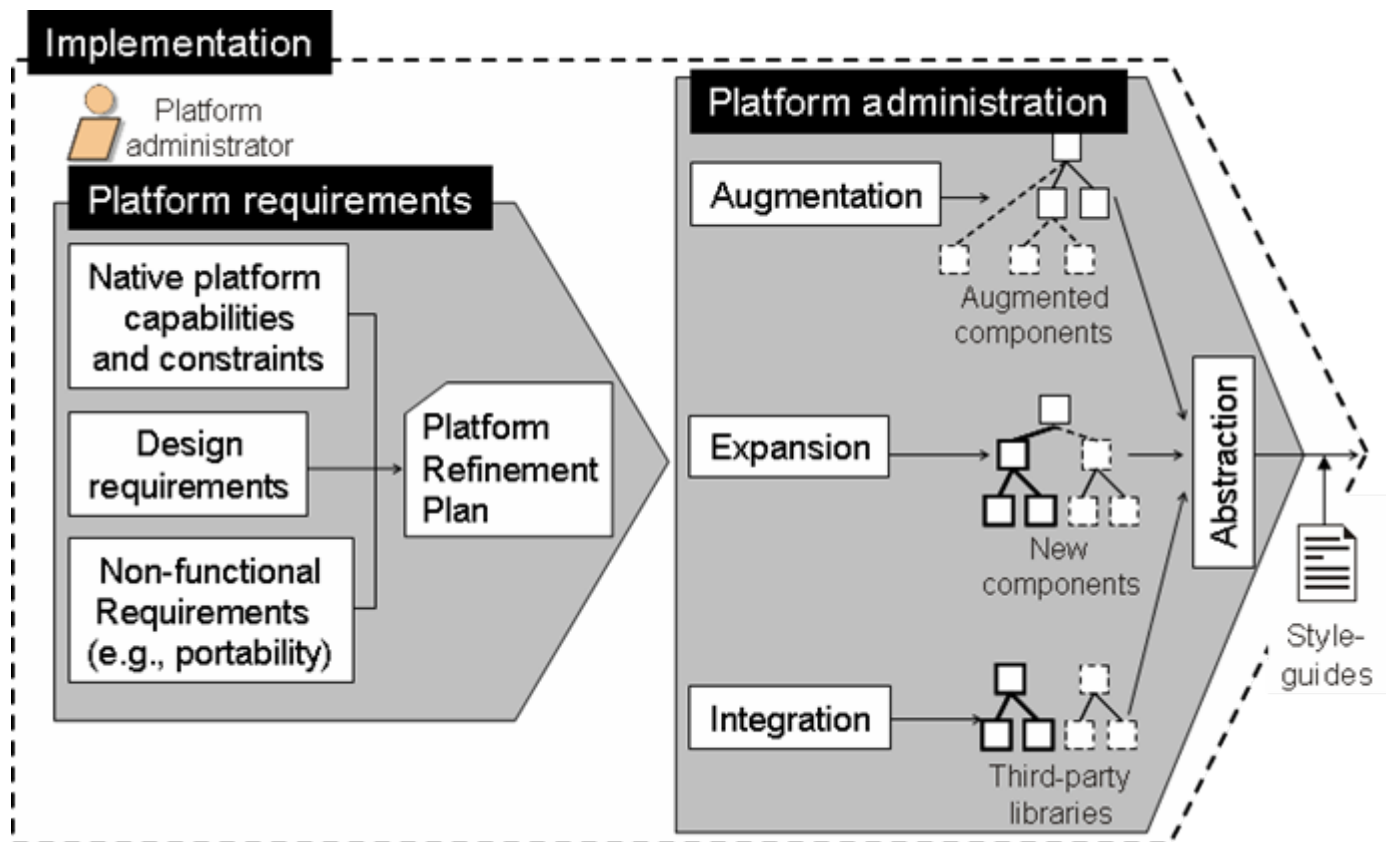
Overview of the strategies

- ■ Augmentation
  - ◈ 'injection' of new interaction techniques in an established library
    - • Same programming model, API, etc
- ■ Expansion
  - ◈ Build domain-specific interaction components to convey metaphoric representation
    - • New top-level container with layout management functions
- ■ Integration
  - ◈ Interoperability with a third-party library
    - • Synchronization of augmented and expanded components with components implemented using the third-party library
- ■ Abstraction
  - ◈ Separate specification from physical implementation
    - • Work currently in progress

Augmentation

- Augmented TabbedPane
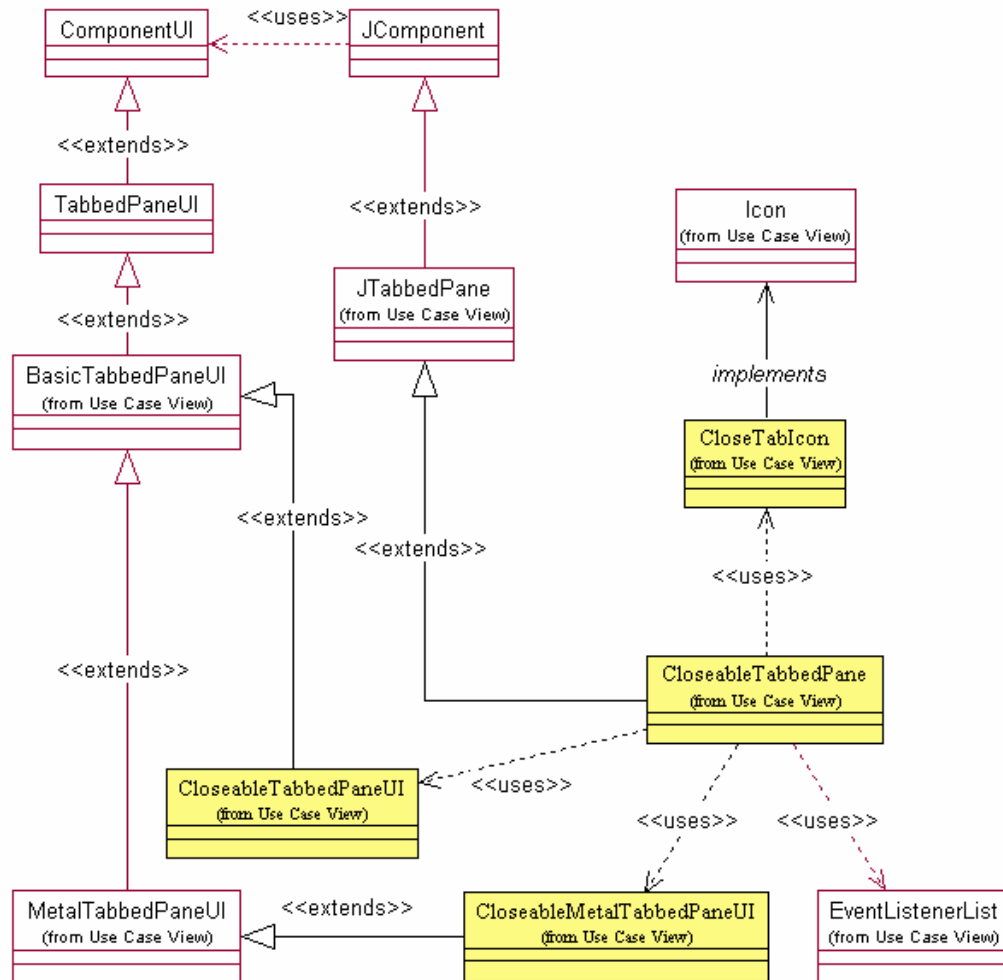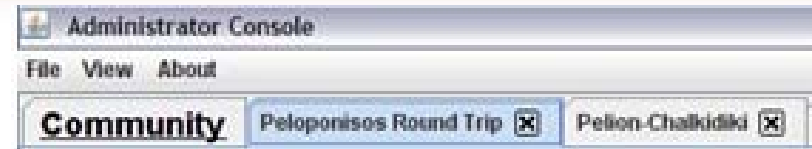
Augmentation (Cont.)

- RadioCheckBoxTree
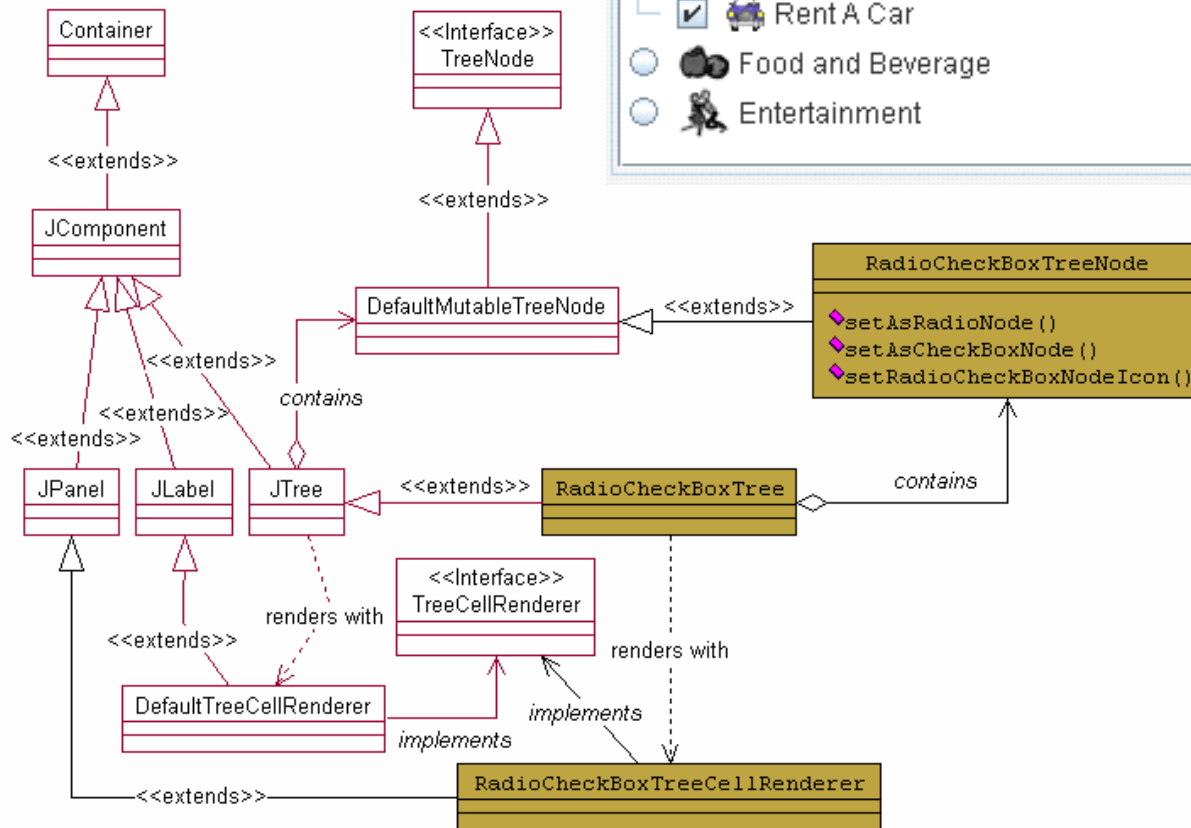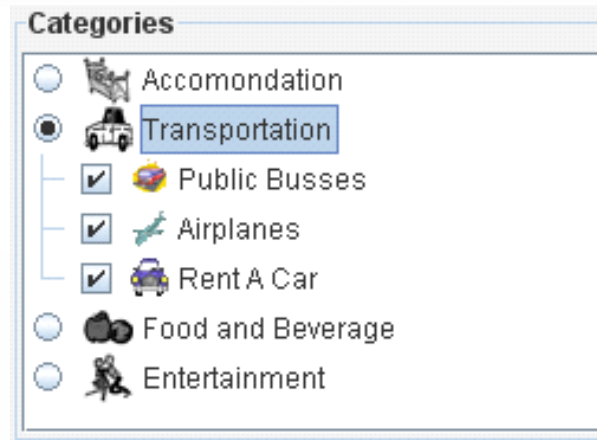
Expansion

- Activity panel = TV program -like
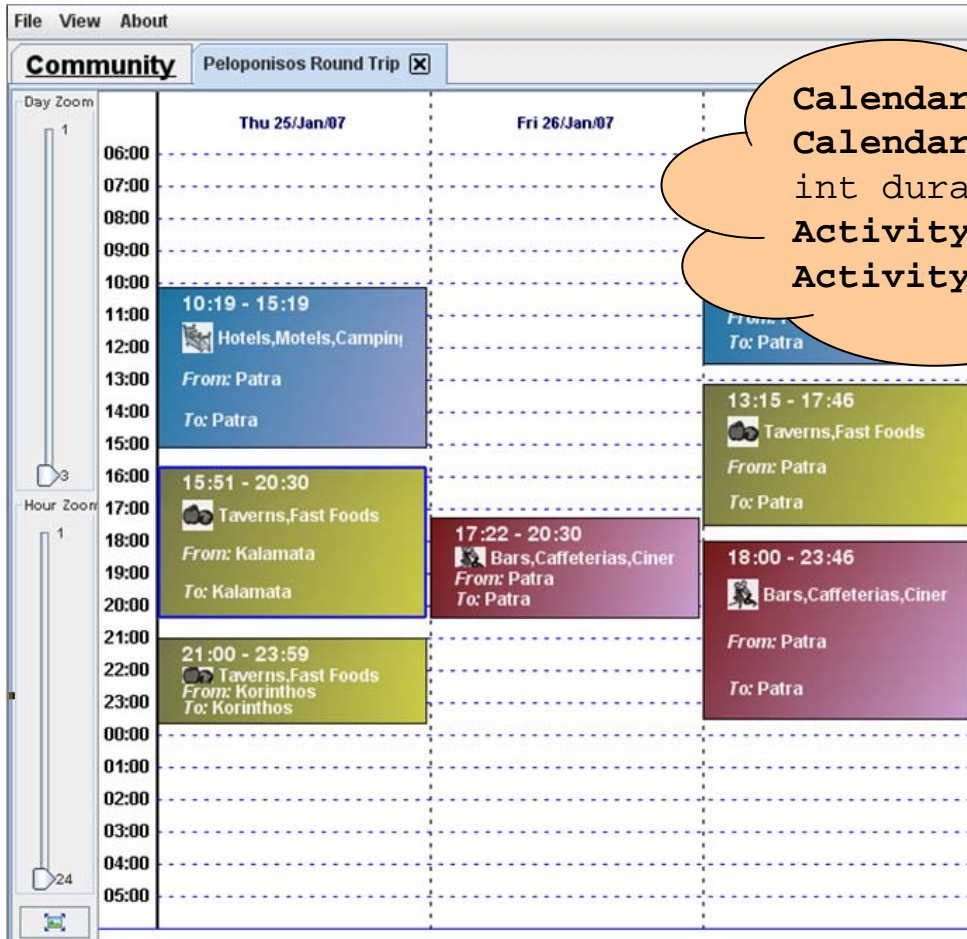

```
Calendar sCal =
Calendar.getInstance();
int duration = 5;
ActivityPanel panel = new
ActivityPanel(sCal, duration);
```

Expansion (Cont.)


```
Activity anActivity = new Activity();
panel.addActivity(anActivity);
panel.removeActivity(anActivity);
```
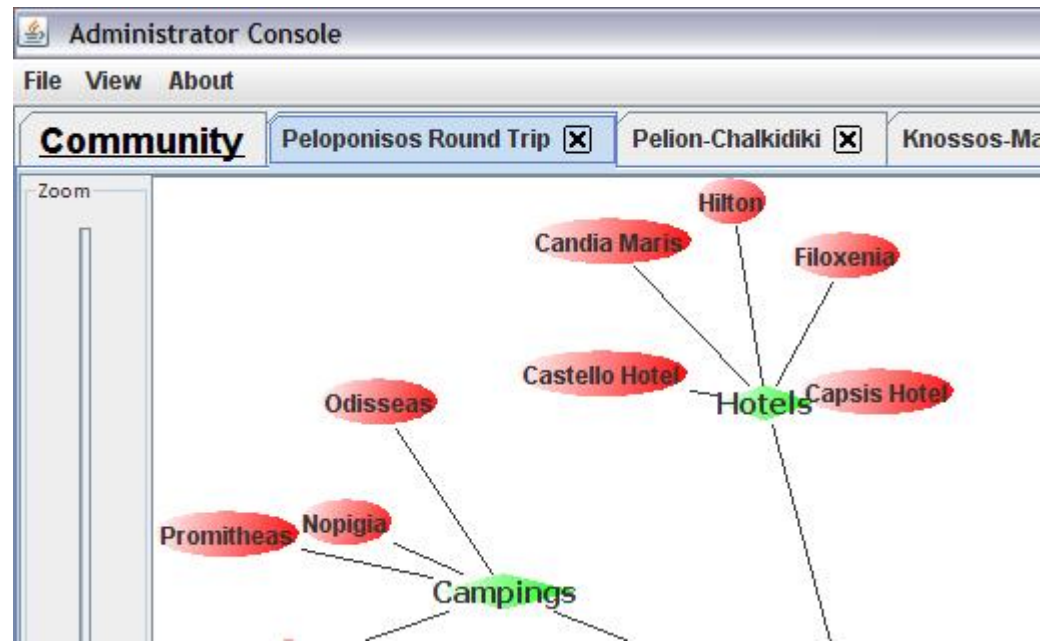
Integration
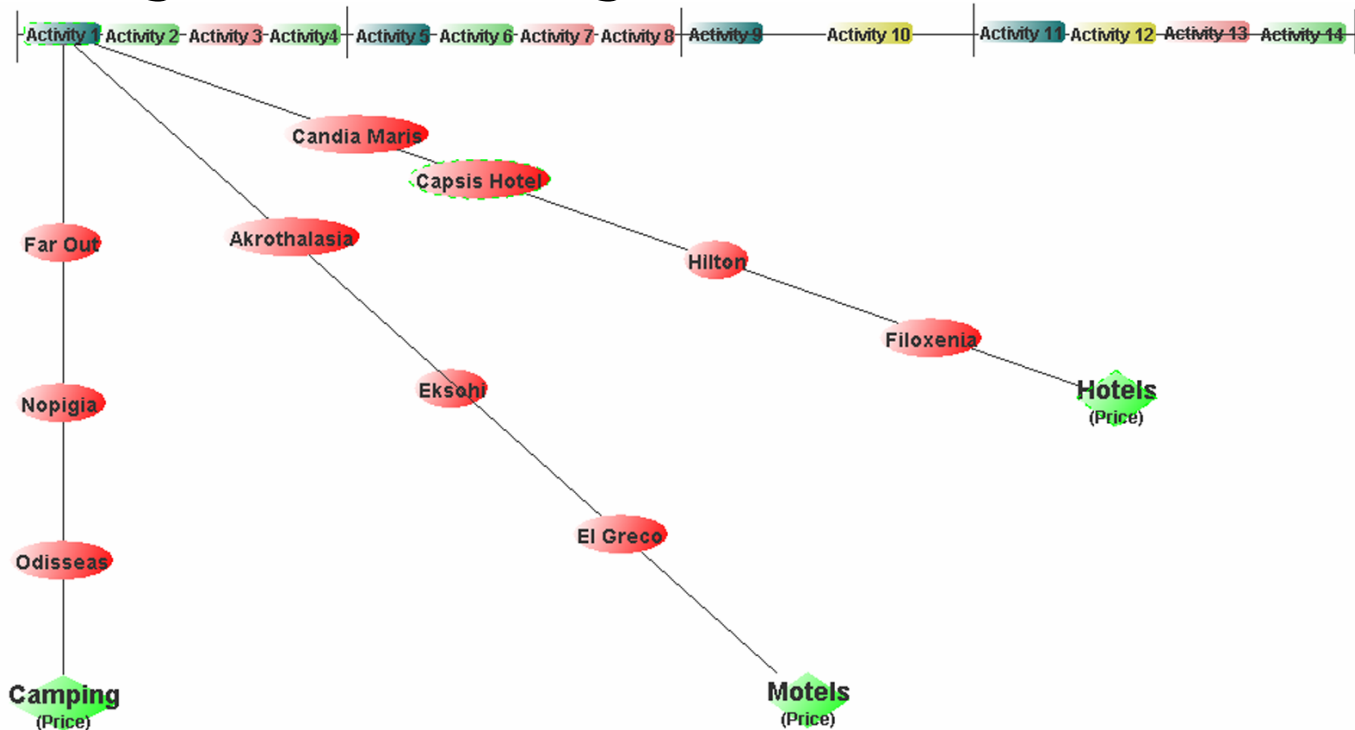
- JGraph (visualization library)
  - Open source http://www.jgraph.com/
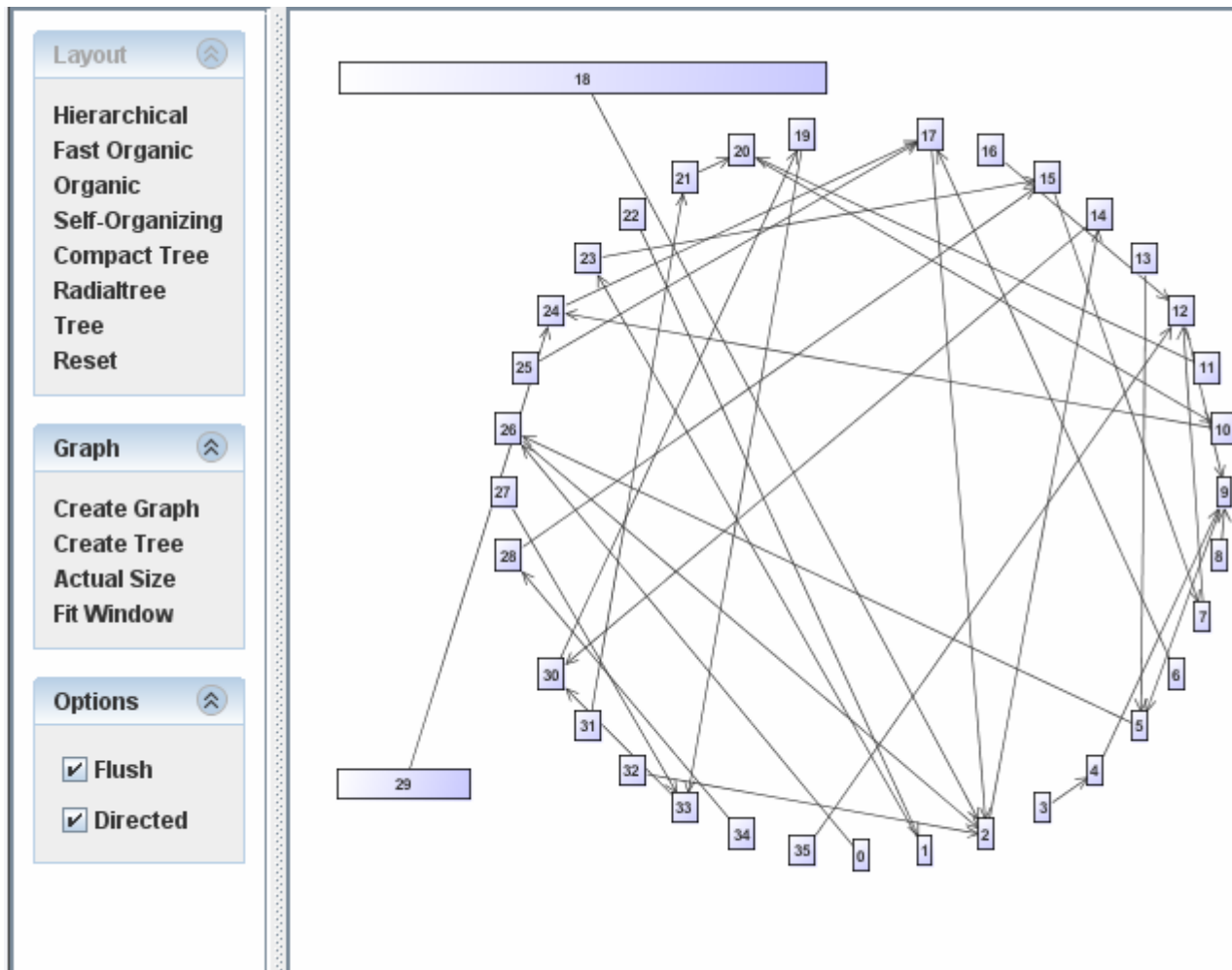
Integration (cont.)

- Refinement of the basic JGraph libraries (visual attributes and layout management) to present generalized containment hierarchies
  - Virtual squad visualization

Integration (cont.)

- Refinement of the basic JGraph libraries (visual attributes and layout management) to present generalized containment hierarchies
  - ✓ Virtual squad visualization
  - Package evolution stages

Integration (cont.)

- Refinement of the basic JGraph libraries (visual attributes and layout management) to present generalized containment hierarchies
  - ✓ Virtual squad visualization
  - ✓ Package evolution stages
  - Shared community message board

Example

- eKoNEΣ project forum data

- Assuming that the squad is the consortium (for demo purposes)
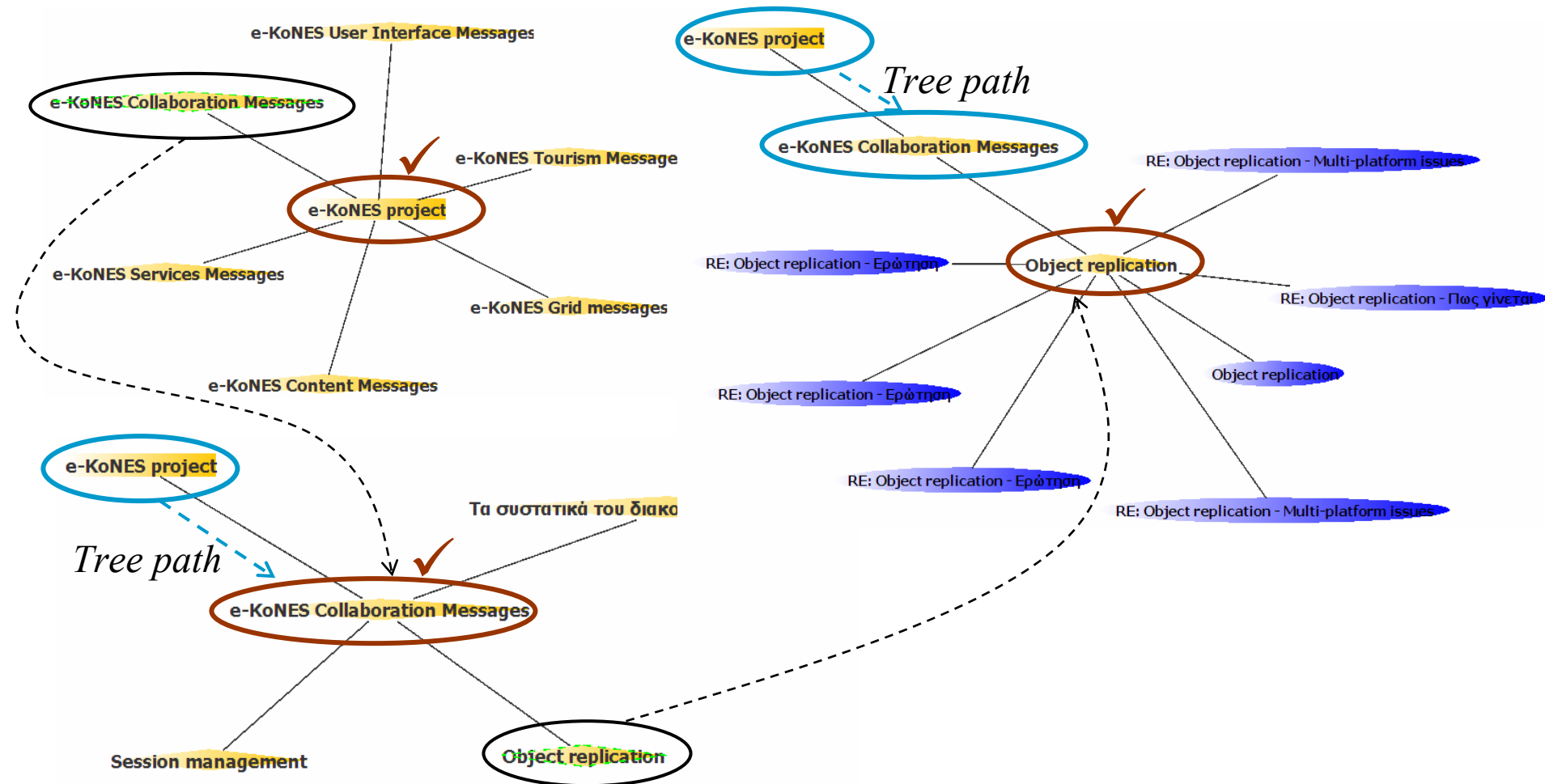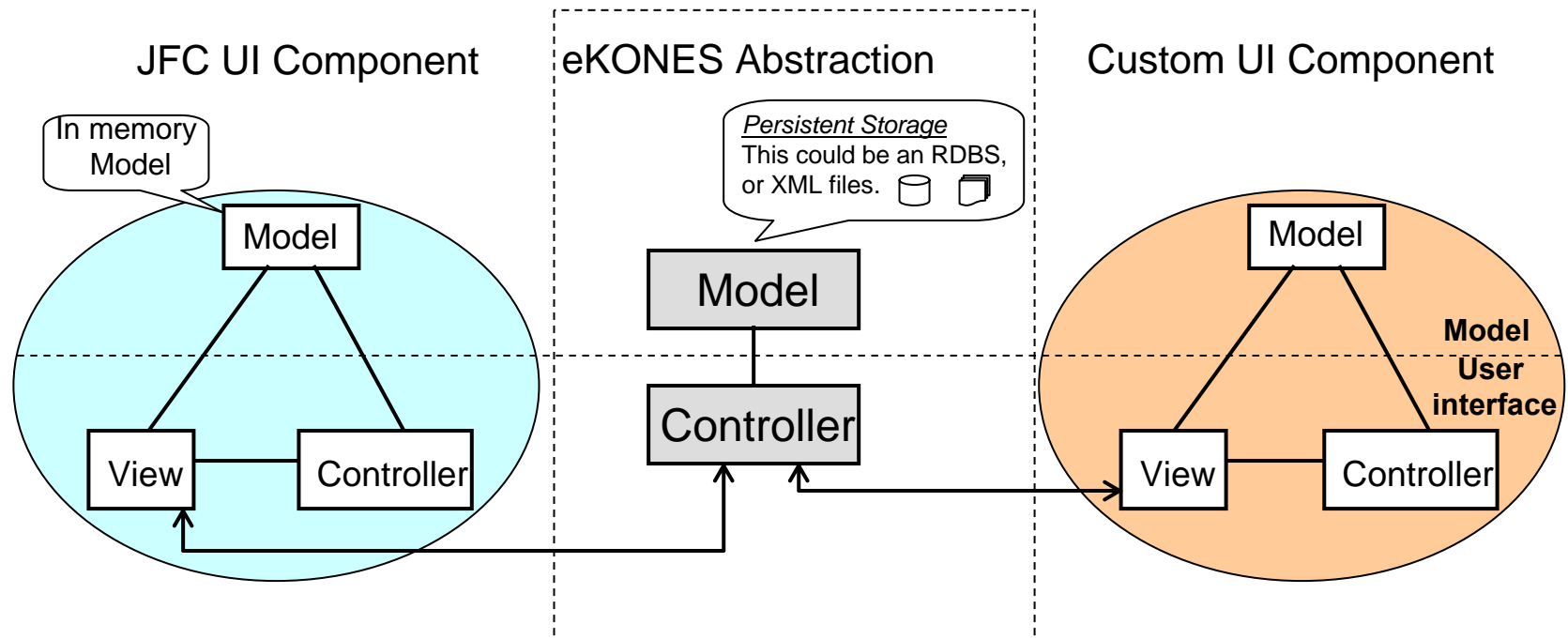
Abstraction

- ## In general, two perspectives
  - ### Abstract components transformed to 'concrete' (platform-specific) interaction objects
    - By transformation rules (e.g., Teresa by Mori, Paterno and Santoro, 2004)
    - By 'hard coded' vocabulary specific mapping rules (e.g. UIML)
  - ### By 'linking' rather than directly 'calling' the libraries of a toolkit
    - PIM tool by Savidis, Stephanidis, Akoumiankis, 1997 in EuroGraphics, DSV-IS Proceedings

Technology server

Abstract component

Swing

MFC
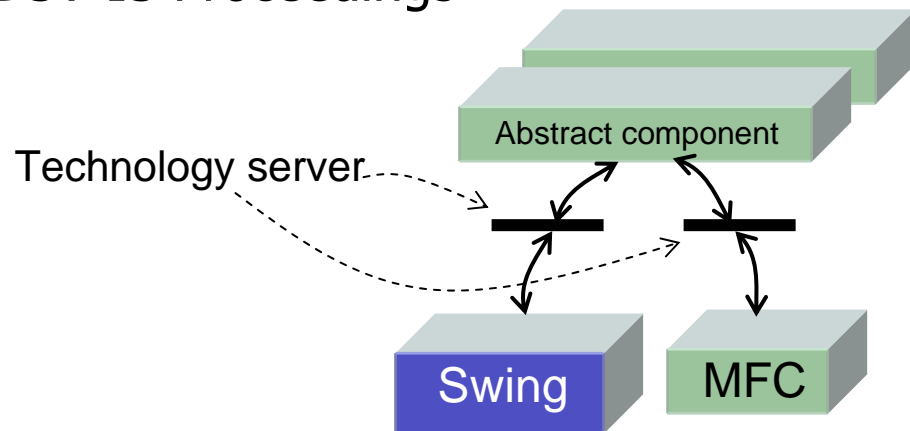
Abstraction (Cont.)

- In eKoNEΣ we have studied a solution exploiting the MVC architecture

- So, what does this allow us to do ?

Abstraction (Cont.)

- Synchronization across views

Summary & conclusion

- So far we have established and pseudo-verified the concept of platform administration in selected scenarios

- On-going work focuses on (relevant issues)
  - Refinement of some of these techniques
    - e.g. abstraction)
  - Extension of the techniques to facilitate
    - Synchronous cross-platform collaboration
    - Multi-user widgets
    - Context-ware widgets
  - Building, visualizing and extrapolating from scalable community memories
    - Virtual squad memory
    - Package memory
  - e-Moderation during collaborative sessions