

# Theme: An Approach for Aspect-Oriented Analysis and Design

Elisa Baniassad and  
Siobhán Clarke

Trinity College, Dublin  
ICSE '04

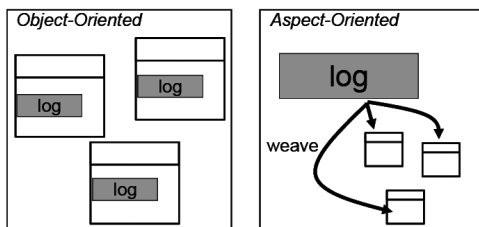
1

## Outline

- Introduction
- Theme
- Views
- Case Study
- Analysis
- Results & Conclusions

2

## Aspect Oriented Programming



3

## Requirements Documentation

- Aspects woven throughout descriptions
- Difficult to trace an aspect through development lifecycle
- Need to identify aspects early

4

## Current Approaches for Identifying Aspects

- ☞ Look for objects first, then identify tangled behavior as it appears
- ☞ Before design process, scan requirements for aspect-style behavior
- ☞ Apply AO requirements engineering technology and target non-functional requirements as aspects

5

## Theme

- Support for viewing of relationships between all behaviors described in requirements documentation, identifying aspects in requirements and modeling those requirements
- theme - an element of design, a collection of structures and behaviors that represents one feature
- Theme/Doc & Theme/UML

6

## Course Management System

- R1. Students can register for courses.
- R2. Students can un-register for courses.
- R3. When a student registers then it must be logged in their record.
- R4. When a student un-registers it must also be logged.
- R5. Professors can un-register students.
- R6. When a professor un-registers a student it must be logged.
- R7. When a professor un-registers a student it must be flagged as special.
- R8. Professors can give marks for courses.
- R9. When a professor gives a mark this must be logged in the record.

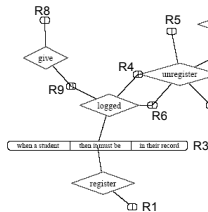
7

## Theme/Doc Action View

- Identify cross-cutting behaviors
- Two inputs needed:
  - List of key actions from developer
  - Requirements
- Theme/Doc performs lexical analysis of text and generates Action View

8

## Theme/Doc Action View



- No actions are isolated from the rest
- Identify aspects by examining shared requirements
- Isolate groups of actions and requirements into base and cross-cutting

Figure 1. Action View of CMS Requirements

9

## Theme/Doc Clipped View

- Examine a shared requirement and determine which action it is more appropriately coupled with
  - Clip the link between the requirement and the less-appropriate action
- R3. When a student registers then it must be logged in their record.

10

## Theme/Doc Clipped View

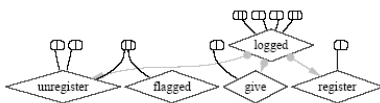


Figure 2. Clipped Action View

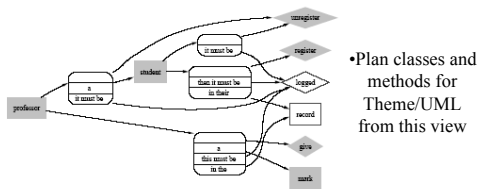
11

## Theme/Doc Theme View

- Used to plan the design and modeling of the themes identified earlier
- Show requirements and actions as well as key elements of system important for theme design in Theme/UML
- Key entities

12

## Theme/Doc Theme View



•Plan classes and methods for Theme/UML from this view

Figure 5. Theme/Doc Theme View: *logged*

13

## Theme/UML

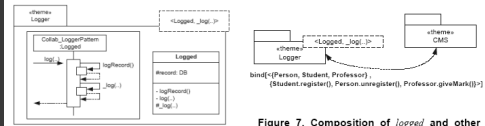
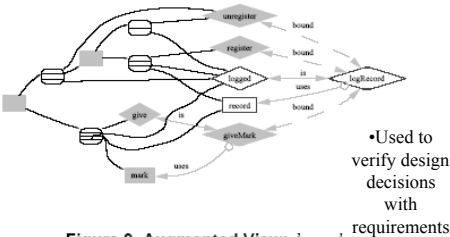


Figure 6. Theme/UML: *logged*

Figure 7. Composition of *logged* and other themes

14

## Theme/Doc Augmented View



•Used to verify design decisions with requirements

Figure 9. Augmented View: *logged*

15

## Case Study – Location Aware Game

- 89 requirements
- Collect crystals deposited throughout a location
- Encounter other players, human and computer generated
  - Interact and perhaps duel with other players
- Winner has the most crystals at the end of a specified time period

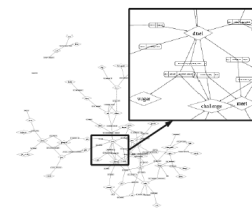
16

## LAG – Finding Themes

- 59 actions identified
- Not all actions need to be separate themes
- Examine action view to determine relationships and larger themes
- Location of actions in graph aided in grouping into themes
  - Actions that share requirements are close to one another

17

## LAG – Theme/Doc Action View



•Examine requirements shared, consider meaning of actions, how should actions be grouped?

•Synonymous actions

Figure 10. Game Action View: All Actions

18

## LAG – Theme/Doc Clipped View

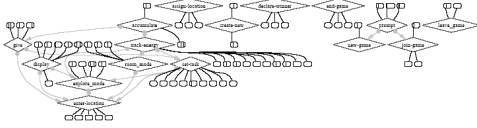


Figure 11. Clipped Action View of Major Game Actions

19

## LAG – Theme/UML

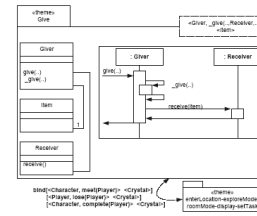


Figure 12. Theme/UML: give

- give theme handles when a player or character gives a crystal to another player
1. Player meets a character
  2. Player loses to another player
  3. Player completes a task for a character

20

## LAG – Theme/Doc Augmented View

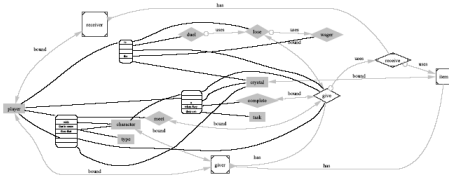


Figure 13. Augmented View: give

21

## Analysis

- Effectiveness
  - Could all aspects have been identified only based on requirements document?
- Requirements coverage
  - Any orphaned requirements?
- Traceability
  - Is requirements document easily linked to design model?
- Scalability
  - What are appropriate "zoom-levels"?

22

## Results

- Synonyms – ex. Give
  - Action view placed seemingly related actions closer together
- Scattered requirements
- Ambiguities in requirements
  - No specific description of a player picking up a crystal when it was discovered in a location
- Evolution of requirements
  - Changes could be made during requirements gathering or after modeling has begun

23

## Conclusions

- Theme effective in helping identify aspects in requirements specifications
- Provides traceability links from requirements to design as modeled in Theme/UML
- Check coverage of requirements in design
- Possibility to enhance scalability

24

## Resources

- <http://www.cs.tcd.ie/Elisa.Baniassad/research.html>