7 Software Engineering Techniques for Multimedia Software

7.1 Design Patterns: The Idea
7.2 Patterns for Multimedia Software
7.3 Gang-of-Four Patterns Applied to Multimedia
7.4 Modeling of Multimedia Applications

Literature:
Gamma/Helm/Johnson/Vlissides: Design Patterns, Addison-Wesley 1994
(= „Gang of Four“, „GoF“)
Design Patterns

• A design pattern is a generic solution for a class of recurring programming problems
  – Helpful idea for programming
  – No need to adopt literally when applied
• Origin:
  – Famous book by Gamma/Helm/Johnson/Vlissides (“Gang of Four”)
    » List of standard design patterns for object-oriented programming
    » Mainly oriented towards graphical user interface frameworks
    » Examples: Observer, Composite, Abstract Factory
• Frequently used in all areas of software design
• Basic guidelines:
  – Patterns are not invented but recovered from existing code
  – Pattern description follows standard outline
    » E.g.: Name, problem, solution, examples
Window Place: Architectural Pattern

Christopher Alexander et al., A Pattern Language, 1977
(quoted in Buschmann et al. 1996)

• **Problem:** In a room with a window and a sitting opportunity users have to decide whether to have a look or to sit.

• **Solution:**
  At least one window of the room shall provide a sitting place.

• **Structure:**

Each pattern describes a problem which occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice.

*Christopher Alexander et al., A Pattern Language*
Description of a Design Pattern

• Name
• Problem
  – Motivation
  – Application area
• Solution
  – Structure (class diagram)
  – Participants (usually class, association und operation names):
    » Role name, i.e. place holders for parts of implementation
    » Fixed parts of implementation
  – Collaboration (sequence of events, possibly diagrams)
• Discussion
  – Pros and cons
  – Dependencies, restrictions
  – Special cases
• Known uses
7 Software Engineering Techniques for Multimedia Software

7.1 Design Patterns: The Idea
7.2 Patterns for Multimedia Software
7.3 Gang-of-Four Patterns Applied to Multimedia
7.4 Modeling of Multimedia Applications
Patterns for Multimedia Software

- The following catalog of patterns is not taken from literature, but derived from the material in this lecture
  - Work in progress, needs to be revised/completed
- Types of patterns:
  - Cross-platform patterns
  - Patterns specific for a certain platform (e.g. Flash, Pygame, JavaFX)
Cross-Platform Multimedia Pattern: Event Handler

- Program code is not executed sequentially but triggered by events
- Space usage: any
- Time usage: Interaction dependent
- Interactivity: any
- Examples:
  - ActionScript event handlers
  - Lingo event handlers
  - JavaFX event handlers
  - Python event handlers
  - ...
Cross-Platform Multimedia Pattern: Clockwork

- The current properties of presentation elements are derived from the current value of a “clock” ticking at regular time intervals
- Time usage: Linear progress
- Limited interactivity: Automatic or confirmations & questions
- Usually combined with static layout or scenes and objects
- Examples:
  - Timeline in Flash, Director
  - EnterFrame-Events in Flash ActionScript
  - Ticking scripts in Squeak
  - PActivity in Piccolo

```java
PActivity flash = new PActivity(-1, 500, currentTime + 5000) {
    protected void activityStep(long elapsedTime) {
        ...
    }
}
```
Cross-Platform Multimedia Pattern: Interpolation

- A parameter (usually regarding a graphical property) is assumed to change its value continuously dependent of another parameter (e.g. time). The dependency can follow a linear or other rules of computation.
  - Fixed values for the dependent parameter are given for certain values of the base parameter.
  - Intermediate values of the dependent parameter are computed by interpolation.

- Space usage: scenes & objects mainly
- Time usage: Linear progress only
- Usually combined with low interactivity (on this level)
- Examples:
  - Tweening in Flash
  - Animation methods in Piccolo
  - JavaFX interpolators

```java
PActivity a1 = aNode.animateToPositionScaleRotation(0, 0, 0.5, 0, 5000);
```
Cross-Platform Multimedia Pattern: Scene Graph

- Graph structure comprises all represented objects together with the operations (transformations) applied to them
- Space usage: Scenes&objects or fully dynamic
- Time usage: Linear progress or interaction dependent
- Examples:
  - Scene graph of JavaFX
  - Scene graph of Piccolo
  - Implicit: Film Explorer view in Flash

“SceneBeans”
Multimedia Pattern for Selected Platforms: 
Player Component

• For standardized time-dependent media types, a pre-fabricated component is made available which provides
  – Playback of associated media files
  – Standard VCR-style controls (play, pause, stop, rewind)
• Space usage: any
• Time usage: Linear progress
• Interactivity: Interactive controls
• Examples:
  – Flash FLVPlayer component
  – JMF Player component
  – QuickTime player in QT4Java

```java
try {
    p = Manager.createPlayer(new MediaLocator("file:"+file));
    p.addControllerListener(new ContrEventHandler());
    p.realize();
}
```