VR-WISE: Conceptual Modeling for Virtual Reality

Prof. Dr. Olga De Troyer
WISE Research Group
Vrije Universiteit Brussel
• Context
• Motivation
Desktop Virtual Reality
- Cheaper and faster hardware
- Dedicated tools
  - Authoring tools
    - 3D Studio Max, Virtools, Google SketchUp, ...
  - Application toolkits (programming libraries)
    - VR Juggler, Java3D, OpenSceneGraph, ...
  - Engines
    - AVOK, Open Dynamic Engine (ODE), Panda 3D, ...
  - Players
    - Octaga, Flux, ...
Second Life
Google Earth
Context

• Virtual Reality on the Internet/Web
  – Increased bandwidth
  – VR standards for the Web
    • VRML, X3D
  – New Web technology, e.g., Ajax3D
  – Increased availability of 3D content
    • e.g., Google 3D Warehouse
Developing a VR application
Developing a VR application

- Requires considerable VR background knowledge
- No systematic development process
  - Informal design phase
  - Lack of methods
  - Lack of abstraction mechanisms
- Expensive
- Time consuming
- A lot of mismatches
VR-WISE: Conceptual Modeling for Virtual Reality

- Context
- Motivation
- Objectives of the research
- Approach
Objectives of the Research

• To open the development of VR-applications to a broader audience
• To allow a domain expert to be more involved in the design of a VR application
• To reduce the overall development time and cost
VR-WISE approach

• Introduction of a Conceptual Design phase
  – High-level intuitive descriptions using the terminology of the application domain
  – Free from VR-implementation details

• Allows
  – Abstracting from implementation issues
    • Reduces the complexity
    • No deep VR knowledge needed
    • Domain experts may be involved
  – Easier and earlier communication with stakeholders
    • Earlier detection of design errors
• Model-based development
VR-WISE Overview

Conceptual Specification Step
- Domain expert
  - Domain Specification
- World expert
  - World Specification

Mapping Step
- Domain expert
  - Domain Mapping
- World expert
  - World Mapping

Generation Step
- VR-application code
• Graphical Conceptual Modeling Language
  – High-level concepts for modeling
    • Concepts and Objects
      – Including complex connected objects by means of joins
    • Positioning of concepts/objects
    • Behavior of concepts/objects
    • (Interaction)
Concepts and Objects

- **Concepts**
  - Domain concepts
    - Building, Pine Tree, Road Sign, Streetlight, Fountain, ...
  - Properties
    - Visual: high, depth, material, ...
    - Non-visual: price, owner, ...

- **Objects**
  - Instances of concepts
    - several Pine Tree instances

Diagram:
- PineTree
- PineTree:smallPineTree
- PineTree:myPineTree
- PineTree:bigPineTree
• **Spatial relations**
  – Objects can be *positioned relative to each other* by means of spatial relations instead of using exact coordinates
    • *in-front-of, above, left-of, ...*
  – More intuitive for non-VR-experts
  Example:
  ```
  my red car *is* 1 meter *in front* of my house
  ```
Positioning Objects

• Orientation relations
  – To orient objects relative to each other by means of their sides
    • left, right, front, back, top, bottom

Example:
my red car’s right side is oriented towards the front side of my house
Complex Connected Objects

- Objects can be connected in different ways
  - Connection axis relation
  - Connection surface relation
Complex Connected objects

- **Constraints** may exist on connections
  - E.g., Hinge constraint for the door
  - Or a Joystick constraint
Complex Object - Example
Complex Object - Example (2)

Components can move along the connection axis:
2.5 to left
2.5 to right

Connection axis is intersection of:
horizontal plane translated Base.Height/2 to bottom perpendicular plane

Connection axis is intersection of:
horizontal plane translated Rail.Height/2 to top perpendicular plane
Behavior

• Specifying behaviors
  – Action-oriented approach
  – Independent from the static properties of the objects and independent of how the behavior is invoked

• Specifying the invocation of behaviors
  – Using events
• Specifying behaviors
  – Primitive behaviors (actions)
    • To change the position or the orientation of an object
      – move, turn, roll, resize, position, orientate …
– Primitive behaviors (actions)
  • To change the appearance of an object at runtime
    – E.g., transform, construct, destruct, group, ungroup, disperse, combine …
Behavior

– Complex behaviors by combining behaviors by means of operators

Examples:

• Temporal operator for synchronizing behaviors

  ![Temporal operator diagram]

• Lifetime operator

  ![Lifetime operator diagram]
• **Events** are used to specify the triggering of behaviors
  - Time Event
  - Context Event
  - User Event (user interaction)
  - Collision Event (inter-object interaction)
  - Constraint Event
More complex behaviors

– More complexity can be expressed by means of a scripting language

```plaintext
Forward (d m)

|speed| 'fast'
|repeat| 3 time(s)
|variable| assign 1 to i; assign 0 to d
|before| assign 5 * i to d
|after| increment i by 1
```
Mr. Phillip’s famous back kick.
Mr. Phillip’s famous back kick.
Mr. Phillip’s famous back kick.
Generated Behavior
VR-WISE: Conceptual Modeling for Virtual Reality

- Context
- Motivation
- Objectives of the research
- Approach
- Tool support
Tool Support
Tool Support

Graphical interface for making the conceptual models
Tool Support

Translates models into natural language
Tool Support

Allows to specify:
- the domain and the world
- the mappings
Tool Support

Allows to preview the world
Fast prototyping
Tool Support

Code generation
• Ontology-based
  – Domain ontology to describe the world
    • Allows to use terminology of application domain
  – Ontologies as internal knowledge representation

• Semantic Virtual Worlds
  – Use of domain ontologies allows to capture real world semantics
    • Semantic search engines
    • Semantic annotations
VR-WISE: Conceptual Modeling for Virtual Reality

- Context
- Motivation
- Objectives of the research
- Approach
- Tool support
- Conclusions
- Current & future work
Conclusions

• **Conceptual design phase** in the development process of VR application
  – More people can create a VR application
  – More people can be involved
    – Domain experts; other stakeholders
  • Easier to satisfy the requirements and the expectations of stakeholders

  – **Model-based**
    • Code can be generated
    • Less expensive and faster

  – **Ontology-based**
    • Easier to incorporate semantics
    • More usable, allows for semantic search, semantic annotations
Current and future work

• Extending the set of modeling concepts:
  – More primitive behaviors, e.g., coloring, sound, ...
  – Mechanism for combining connections
  – More constraints, like constraints on behavior
  – Cameras, viewpoints, light sources, shadows, ...
  – Interaction-controlled behavior
  – Avatars

• Current work
  – Scenarios
  – Patterns for modeling behavior/the scene
  – Semantic annotations for existing worlds
More information

Part of this work has been done in the context of the **OntoBasis** project and the **VR-DeMo** project (IWT)

See [http://vr-wise.vub.ac.be](http://vr-wise.vub.ac.be)