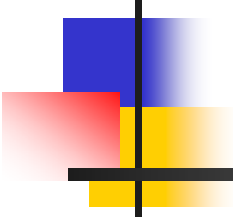


# **A component oriented design for a virtual reality based application**

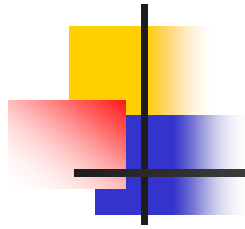


---

by

Michael Haller

Polytechnic University of Hagenberg



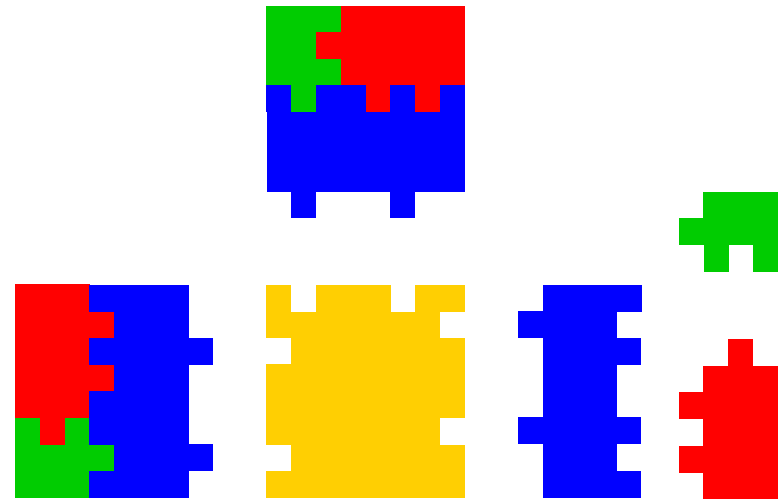
# Contents

---

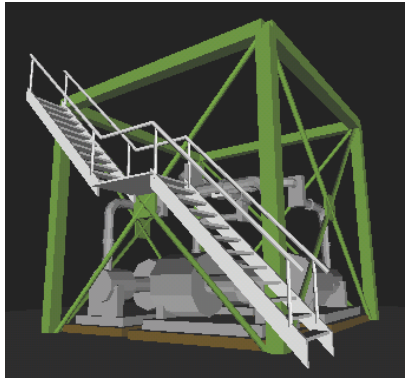
- Motivation
- Component based architecture
- Implementations
- Modeling tools
- SAVE (Safety Virtual Environment)
- Future

# Motivation

- Real-time virtual environments
- Virtual environments are very complex
- Very heterogeneous: different hard- and software systems
- Programming knowledge and scripting languages
- Goal: simple component oriented concept
- "LEGO"-Kit



# Motivation: OMVR->SAVE



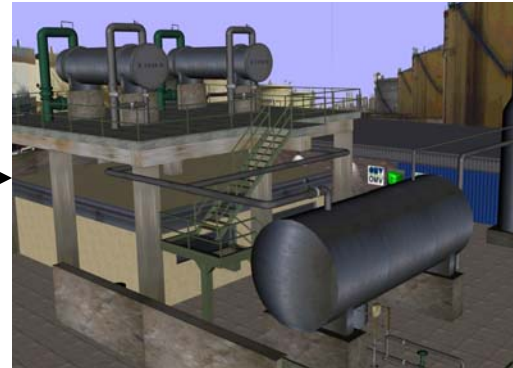
omVR/1 (1997)

1<sup>st</sup> Prototype  
feasibility study  
static objects



omVR/2 (1998)

training scenario H<sub>2</sub>S  
HMD, SGI O2



SAVE (2000)

Different scenarios  
Dynamic and static  
objects

?

SAVE (2001)

Modeling tools,  
motion platform

New framework for a more  
flexible design of our virtual  
environment



# Requirements for the design

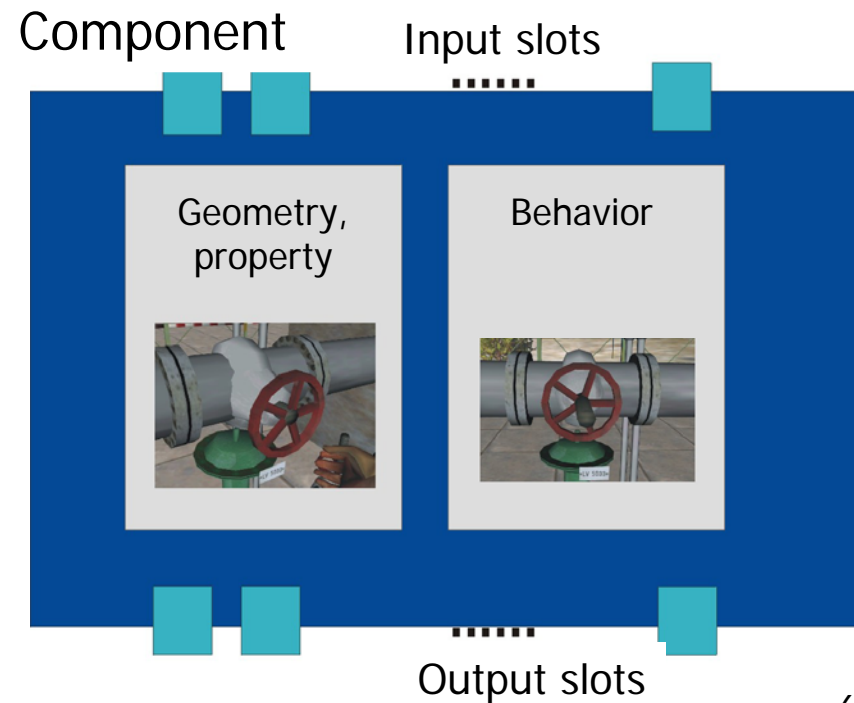
---

- Flexibility
- Reusability
- Extensibility
- Usability
- No scripting languages
- Easy to learn

Everybody should be able to build a virtual environment

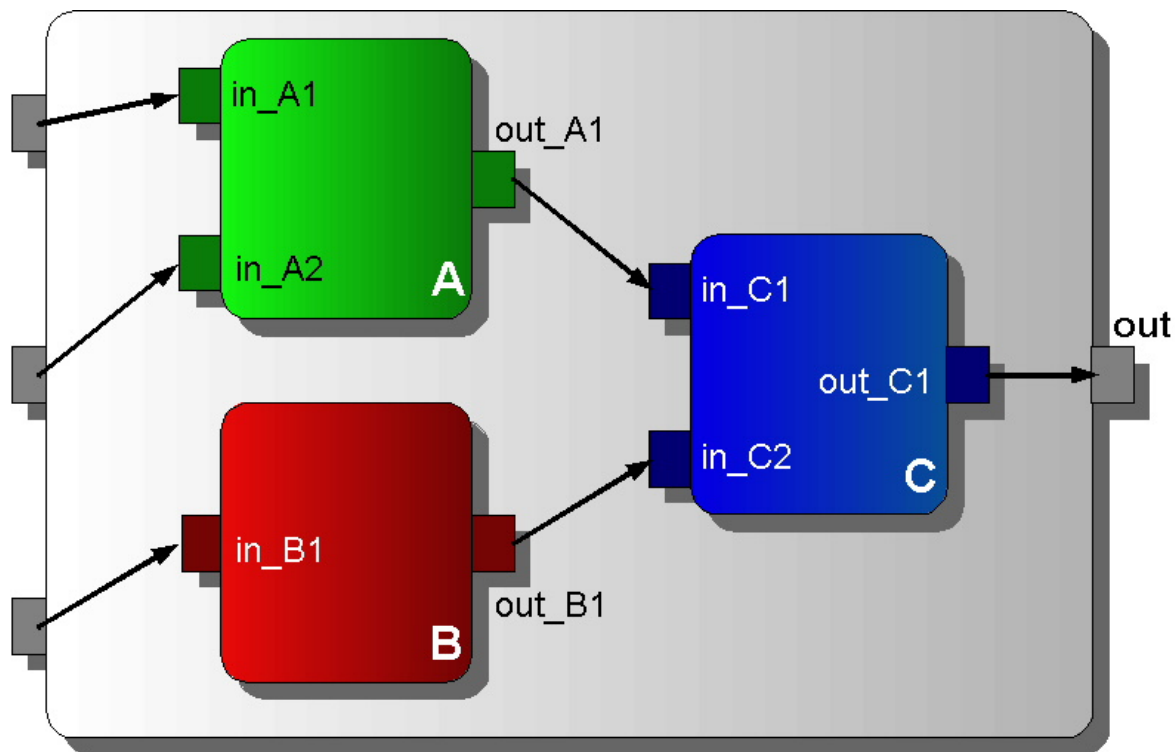
# Component oriented design

- Comparable with:
  - Semantic networks and Frames
  - JavaBeans
  - COM
- Modularity
- The component



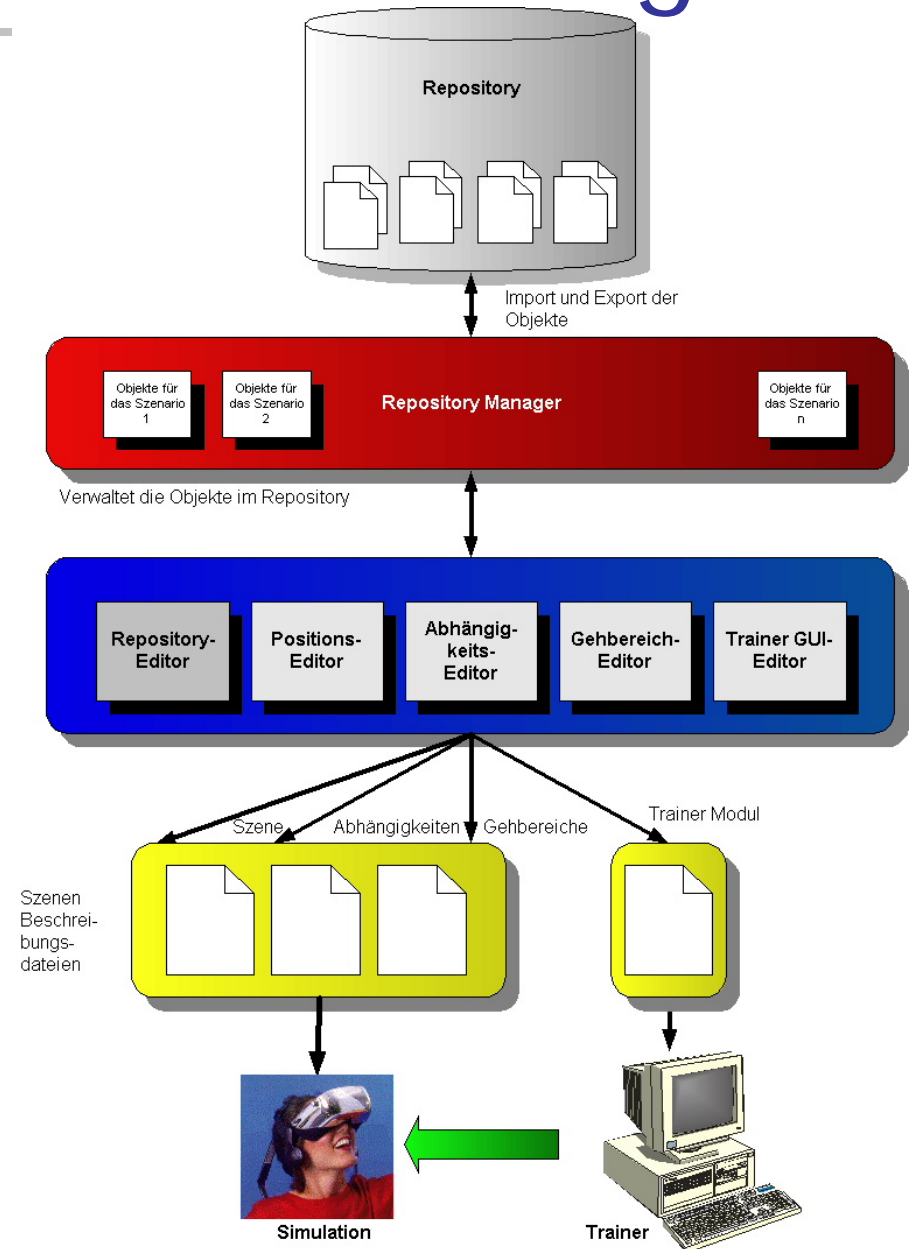
# Component oriented design

- Simple components
- Components have their behavior (complex) and can be connected together
- Compound objects

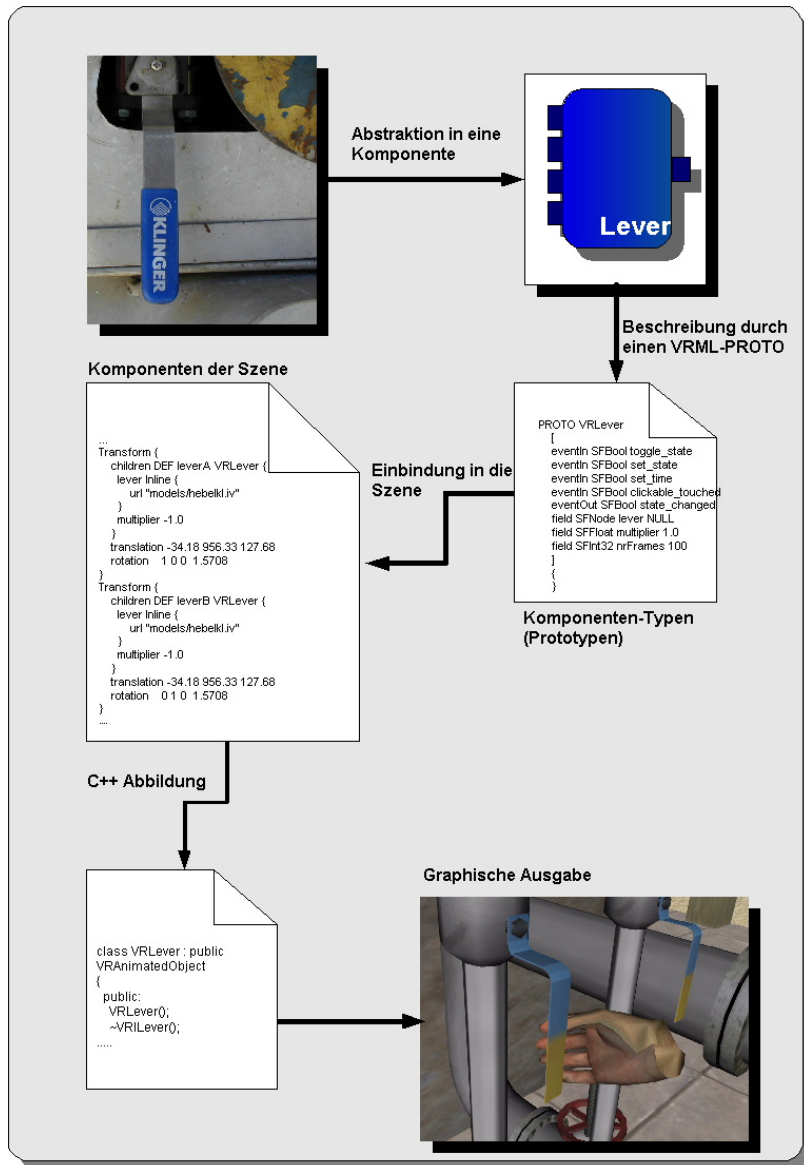


# Component oriented design

- Repository
- Repository Manager
- Modeling tools
- Scenario description file
- Simulation



# Implementations



Real object

*Abstraction*

Component (Lever)

*Mapping*

Prototype

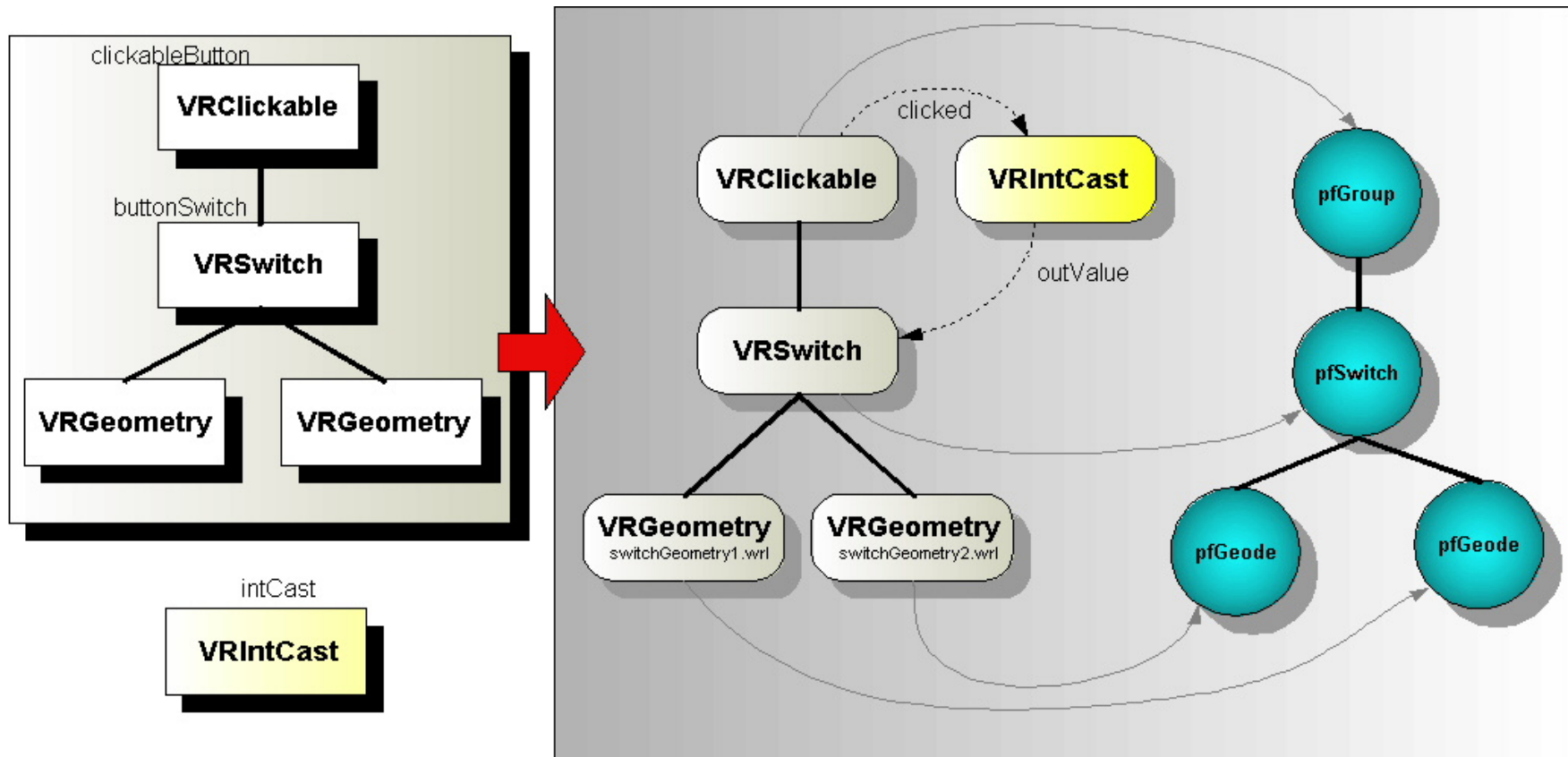
*Intergration*

Scenario description file  
contains the objects

*Implementations*

Simulation

# Implementations



ROUTE clickableButton.clicked TO intCast.inState  
 ROUTE intCast.outValue TO buttonSwitch.setChoice

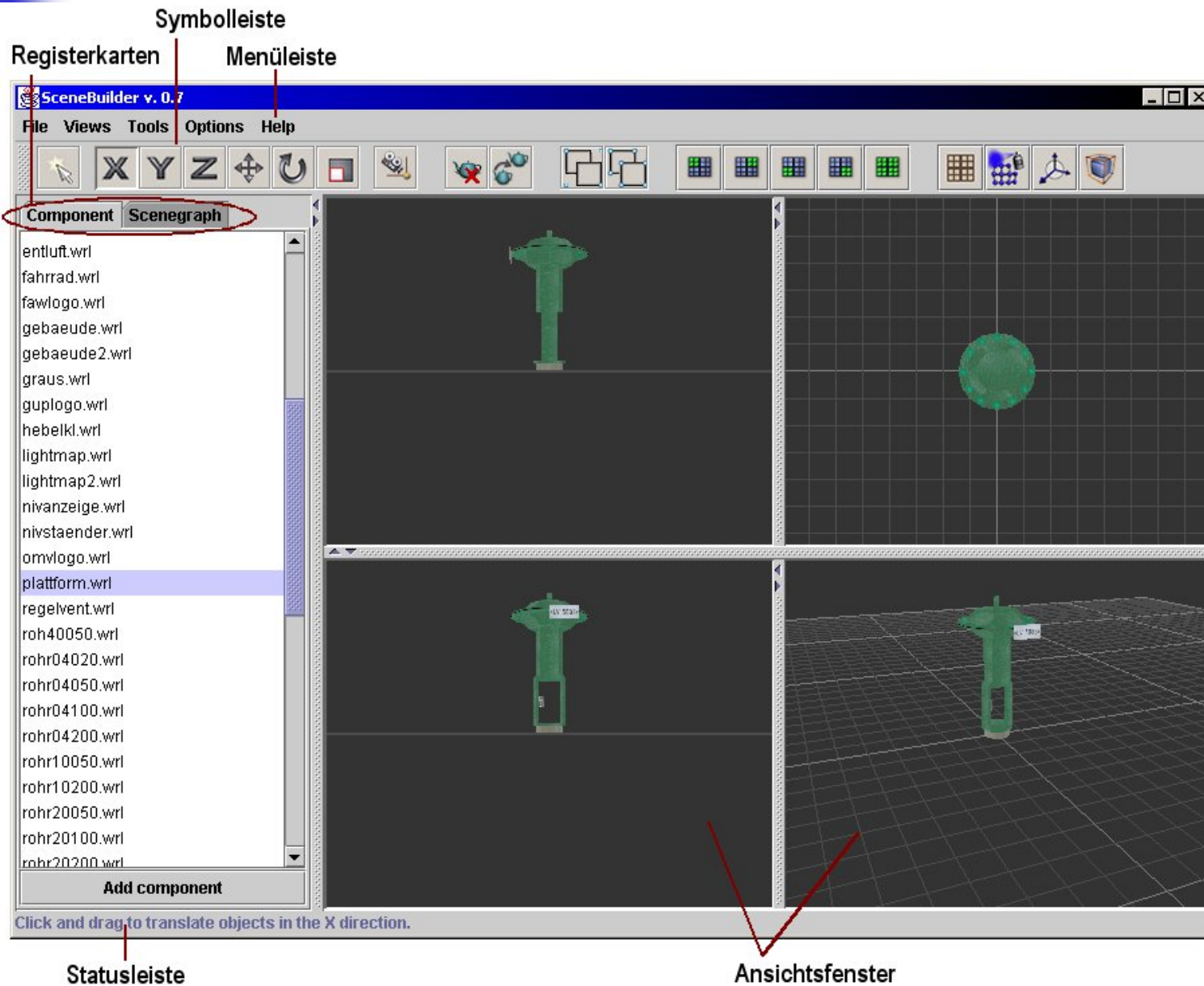


# Modeling tools

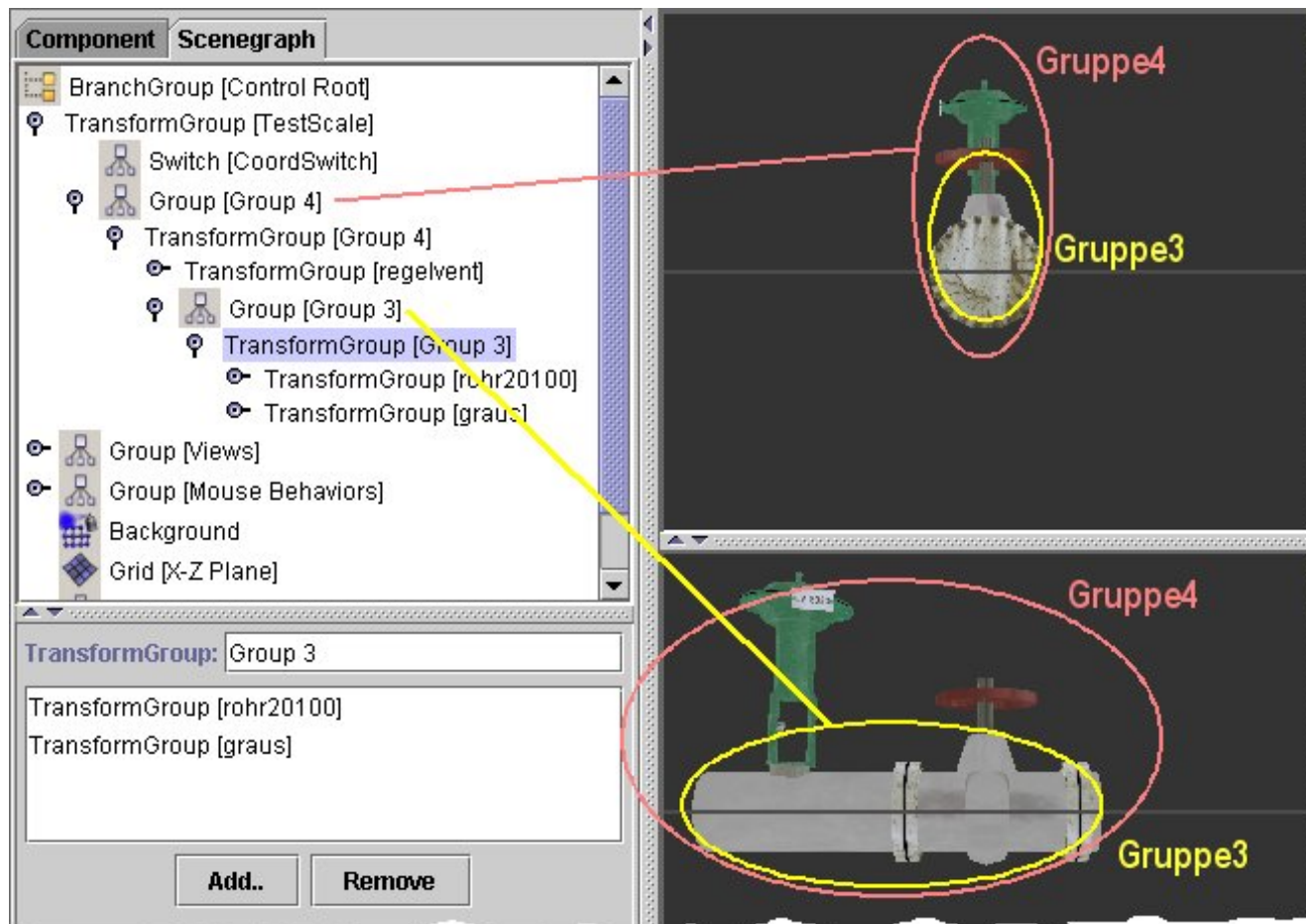
---

- Repository
- Modeling tools
  - SceneBuilder
  - RouteEdit - Dependencies tool
  - ....
- Scenario description file
- Simulation

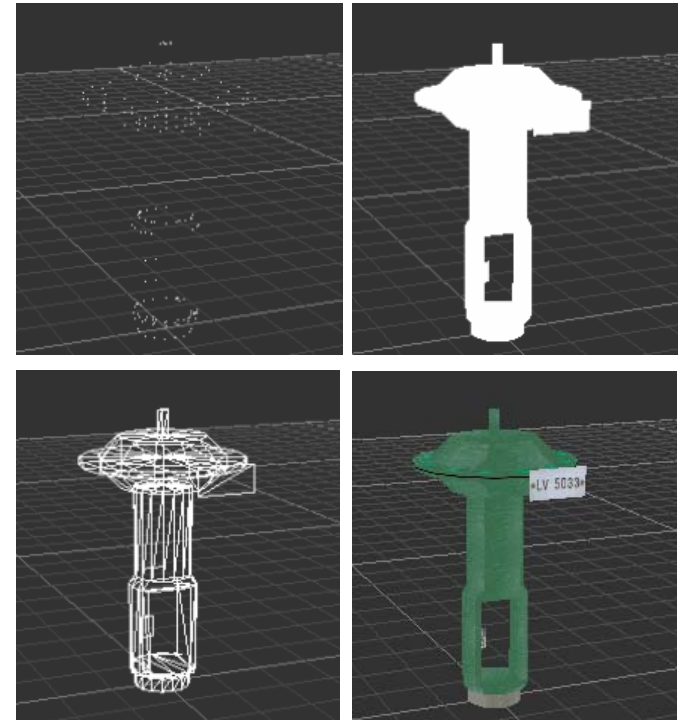
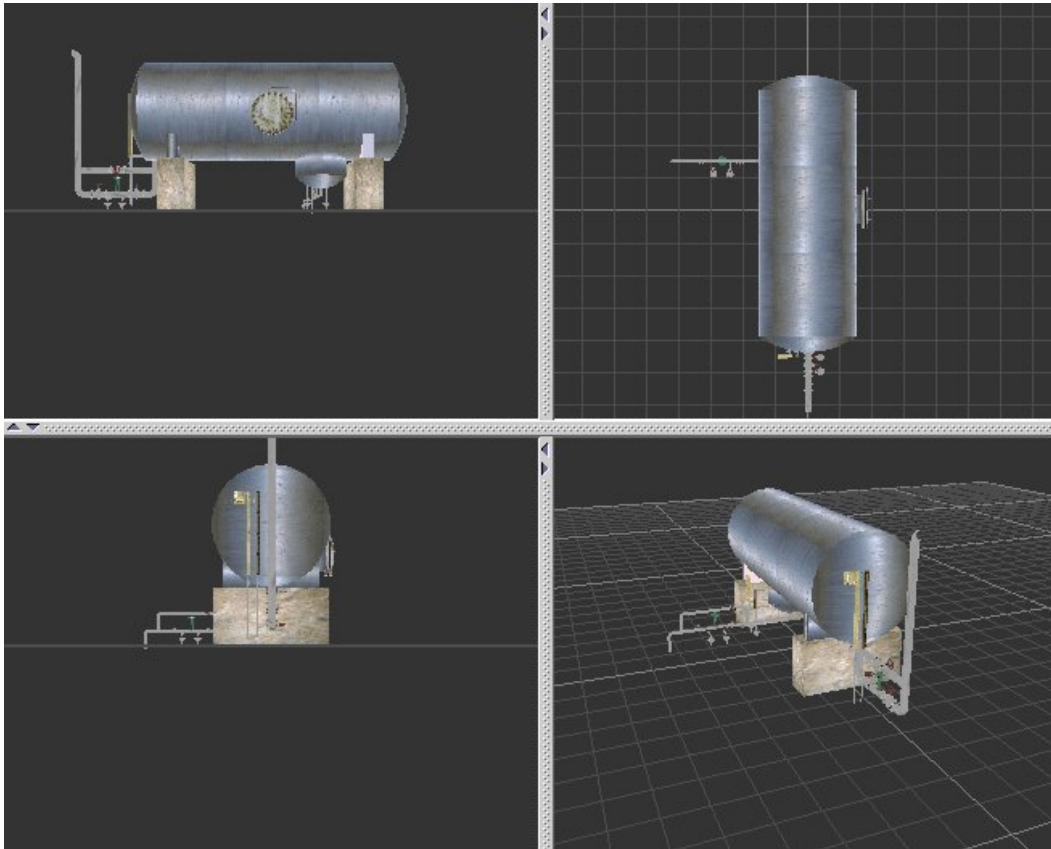
# SceneBuilder



# SceneBuilder



# SceneBuilder

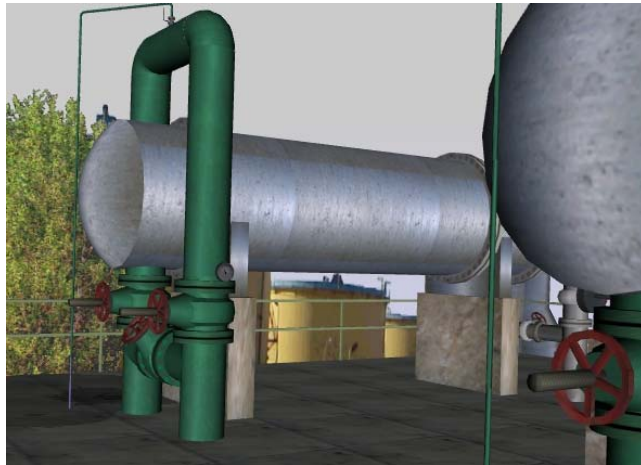


# RouteEdit

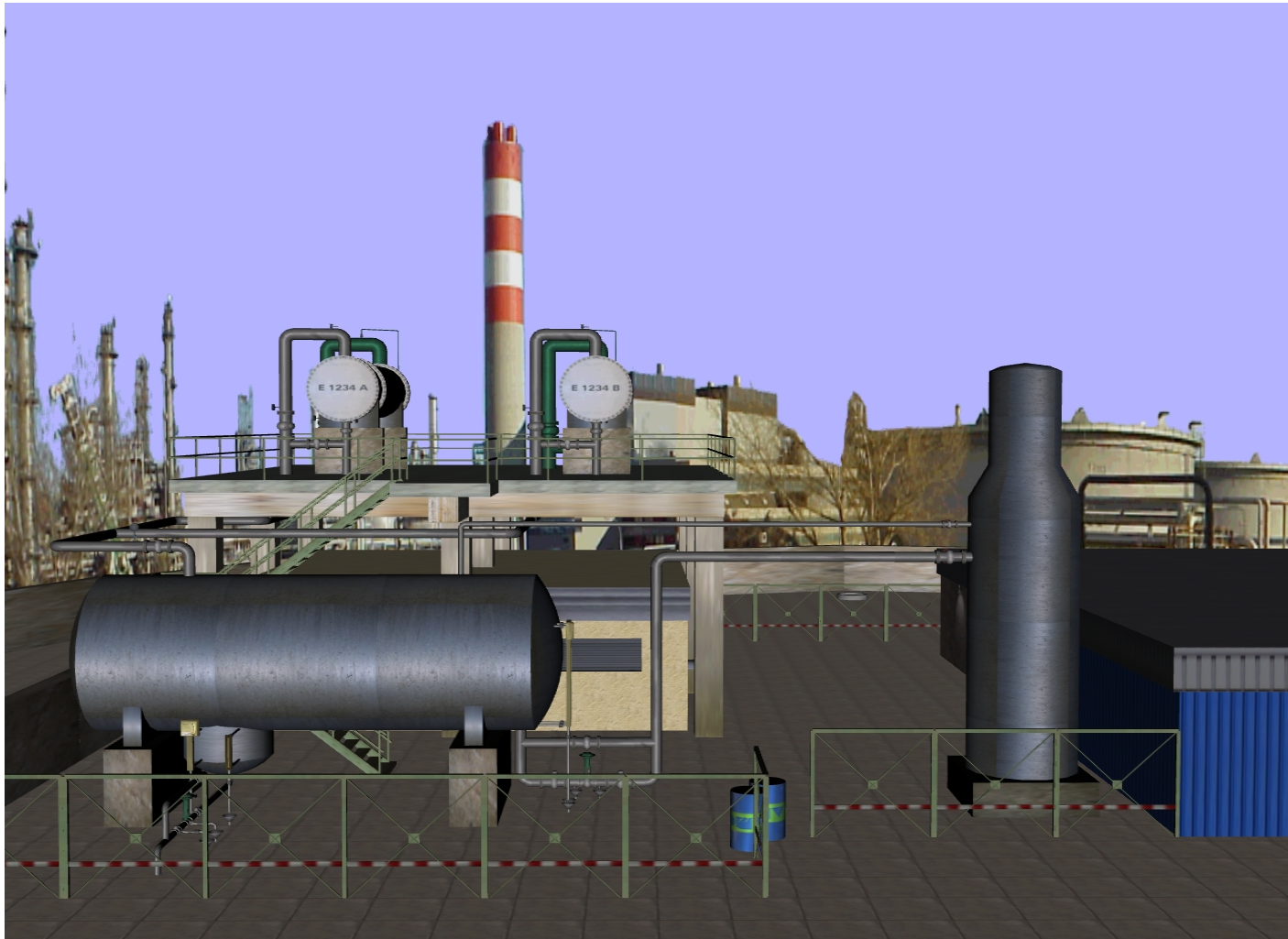
The screenshot displays the RouteEdit application window. At the top, there is a menu bar with 'File', 'Edit', and 'View'. Below the menu bar are four panels: 'Component types' (listing VRMagicButton, VRManometer, VRMax), 'Compound component ty...' (listing VRBypassSchieber, VRTest), 'Components' (listing manometer, max), and 'Route Commands' (containing the text 'ROUTE manometer.value\_changed TO max.set\_in'). An 'Errors:' section is also present. A toolbar with various icons is located below the panels. The main workspace shows a diagram with two component boxes. The left box, labeled 'manometer' and 'VRManometer', contains 'set\_value' and 'set\_time' ports. The right box, labeled 'max' and 'VRMax', contains 'set\_in' and 'out\_changed' ports. A yellow arrow points from the 'value\_changed' port of the 'manometer' component to the 'set\_in' port of the 'max' component.

# SAVE - Safety Virtual Environments

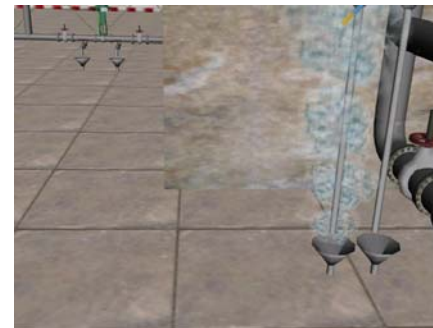
**Safety training for a virtual refinery**



# SAVE - Safety Virtual Environments



# SAVE - Safety Virtual Environments





# Summary

---

- Personnel reviews
  - Component oriented design
    - performance (20-25 fps)
    - Scripting language
  - Modeling tools
    - SceneBuilder
    - RouteEdit
- OpenGL Performer (IRIX)
- Complex components
- More complex components, LOD, PC-version (Linux)