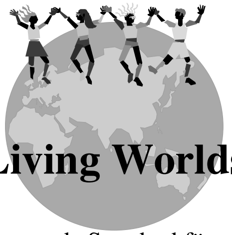
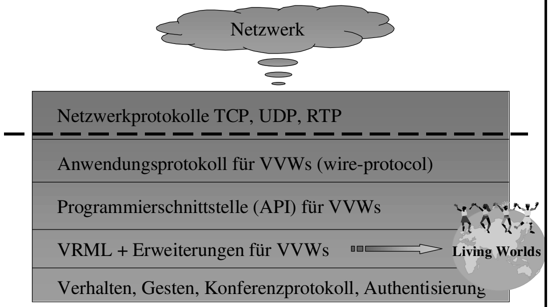


Living Worlds

Der kommende Standard für verteilte, virtuelle Welten

(auf Ebene der Szenenbeschreibung!)

Abstraktionsebenen für verteilte, virtuelle Welten (VVWs)



Entwicklung

- **VRML 1.0:** Aussehen — *Sehen*
- **VRML 2.0:** Verhalten — *Bewegen*
- **Living Worlds:** Scene Sharing — *Teilen*

= Interaktion mehrer Benutzer in auf verschiedenen Rechnern laufenden Kopien **einer** Szene

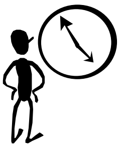
Abstraktionsniveau von LW: Eigenschaften der Kommunikation zwischen den Kopien der Szene und der darin enthaltenen Objekte können deklariert werden und müssen nicht programmiert werden.

Motivation

- Interpersonality, d.h. Mehrbenutzersysteme
- Interoperability, d.h. "Komponenten"
  - Avatare
  - Objekte
  - MUTech
  - Teilwelten
  - Benutzer können Objekte und Avatare in anderen Welten wiederverwenden
  - Entwickler von Komponenten haben größeren Kundenkreis
  - Entwickler von Anwendungen können Komponenten verschiedener Anbieter kombinieren
- Architekturunabhängig: keine Festlegung auf Client/Server oder Peer-to-Peer

Möglichst minimale Schnittstelle

Motivation



Die Entwickler von LW bezweifeln, daß die Zeit reif ist für ein standardisiertes Wire-Protokoll (etwa DWTP oder VRTP), da es noch nicht genügend Erfahrungen aus Anwendungen gibt und die Optimalitätsanforderungen teilsweise widersprüchlich sind.

Vermutung: Motivation könnte auch der Versuch sein, die Investitionen in ihre eigenen, proprietären Wire-Protokolle und APIs zu retten.

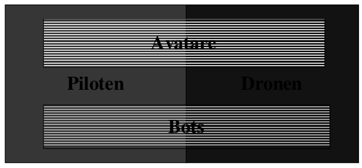
Terminologie

- Eingrenzung und Definition überladener Begriffe
- Festlegung der Abstraktionsebene
- **Räume**
  - **Szene** = Menge von Objekten, die in einem dreidimensionalen Raum angeordnet sind
  - **Welt** = durch Links (Anker) verbundene Menge von Szenen
  - **Zone** = begrenzter Teil einer Szene, enthält verteilte Objekte

Terminologie

- **Lokale Objekte**
- **Verteilte Objekte**
  - **Pilot** = Objekt, das eigenständig Zustand ändert
  - **Drone** = Objekt, das Zustandsänderung mitvollzieht
  - **Avatar** = sein Pilot wird von einem Menschen gesteuert
  - **Bot** (von robot) = sein Pilot ist ein Programm
- **MUTech** (multi-user technology) = Schnittstelle zwischen Browser und Netzwerk. In LW: Komponente in **einer** Zone, die verteilte Objekte verwaltet

Verteilte Objekte

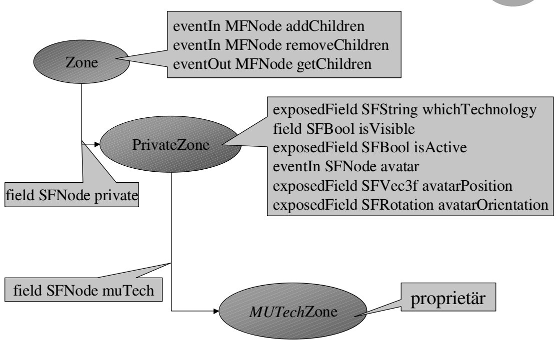


Eine Instanz eines verteilten Objekts kann während seiner Lebenszeit seinen Status verändern, d.h. ein Pilot kann eine Drone werden und umgekehrt, ein Bot kann ein Avatar werden und umgekehrt.

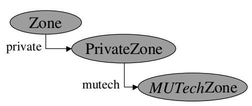
Verhalten von Verteilten Objekten

- Das Verhalten oder die Funktionalität eines verteilten Objekts kann von verschiedenen Komponenten bereitgestellt werden:
  - Avatar/Bot
  - Browser (z.B. Videoeingabe)
  - MUTech (für jede Zone verschieden)
  - Welt (über Zonengrenzen hinweg)

Zonen

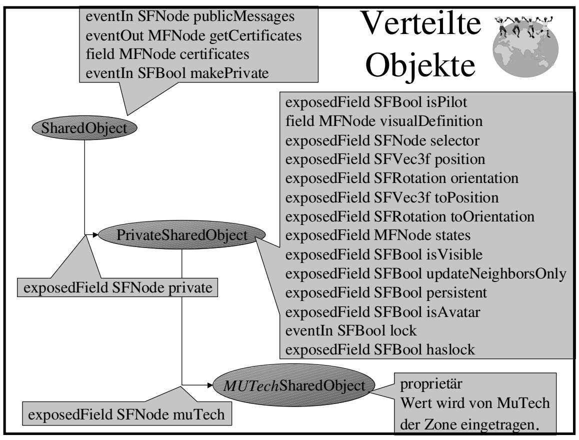


Zonen

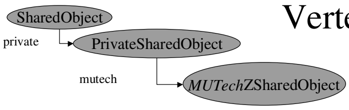



```
DEF Z Zone {
    private DEF PZ PrivateZone {
         zone USE Z
         muTech SomeCompanyZone { private USE PZ
                                             … }
         whichTechnology "SomeCompany"
    }
}

DEF PROX ProximitySensor{ size 100 100 100 }

ROUTE PROX.isActive TO PZ.set_isActive
ROUTE PROX.position_changed TO PZ.set_avatarPosition
ROUTE PROX.orientation_changed TO PZ.set_avatarOrientation
```

Verteilte Objekte



Verteilte Objekte




```
DEF SO SharedObject {
    private DEF PSO PrivateSharedObject {
        visualDefinition
            [ Transform {
               children [
                    Shape {
                        geometry Cone{}
                    }
               ]
               } ]
        sharedObject USE SO
    }
}
```

Nachrichten und Selektoren

```
DEF SC Script { directOutput TRUE
                eventIn SFNode message
                eventOut SFColor setColor
                field SFColor blue 0 0 1
                field SFColor red 1 0 0
                url "javascript: function message(val,t)
                       { if (val->what='Blue') setColor=blue;
                         if ( val->what='Red') setColor=red; }"
               }
DEF SO SharedObject {
    private DEF PSO PrivateSharedObject {
      visualDefinition [
         DEF PERSON Transform { children
            [ Shape { geometry Cone { }
                      appearance Appearance {
                         material DEF M Material {
                          diffuseColor 1.0 0.0 0.0 } } } ] } ]
      pilotSelector …
      states … }
ROUTE SC.setColor TO BODYCOLOR.set_value
ROUTE BODYCOLOR.value_changed TO M.diffuseColor
```

Nachrichten und Selektoren

```
DEF SO SharedObject {
    private DEF PSO PrivateSharedObject {
      visualDefinition …
      pilotSelector
        DEF SEL Selector {
            selectorItems [
                SelectorItem { label "Body Red !!"
                              messageName  "Red"
                              messageScript  },
                SelectorItem { label "Body Blue !!"
                              messageName  "Blue"
                              messageScript  USE SC } ] }
      states [ DEF BODYCOLOR
                    NetworkSFColor{ tag "BodyColor" } ]
      sharedObject USE SO
    }
}
ROUTE SC.setColor TO BODYCOLOR.set_value
ROUTE BODYCOLOR.value_changed TO M.diffuseColor
```

Netnode wie in VSPLUS