A software toolkit for web-based virtual environments based on a shared database

Boris van Schooten, Anton Nijholt, Betsy van Dijk

Parlevink Group

Dept of Human Media Interaction

University of Twente

Netherlands

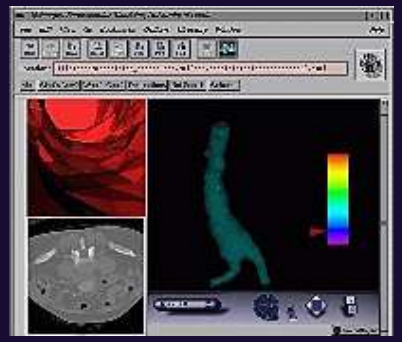
A software toolkit for web-based virtual environments based on a shared database

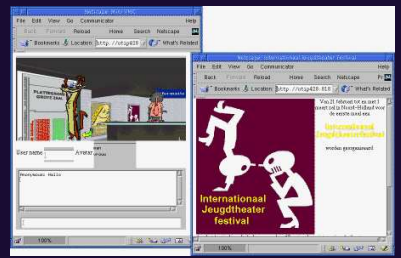
Contents

- Introduction
- Rationale
- The toolkit explained
- Examples
- Conclusions

Introduction

Web-based Virtual Environments (VEs)





Characterisation:

- Information browsing

  ex. Web pages, VR

- Graphical and 3D

  ex. VRML, 3D browsers

- Multi-user (synchronous)

  ex. Chat, online games, web virtual environments

- Interface agents

  ex. mainly web virtual environments

Rationale

Other user interface (UI) architectures:

Component and agent architectures, user interface toolkits (Amulet, VRML, Marigold)

VETk: a shared SQL type database for passing UI information.

Advantages

- Usability
- World structure modelling
- Software engineering

Usability advantages

- **UI structure modelling**.

  A model of the UI structure is useful usability information.

  Like **ERMIA** method, uses **structure diagrams**.

  Additionally: **executability**

- **Prototypability**.

World structure modelling

Software structure ⇔ VE (semantic) structure.

Readily maps to semantic structure as found in various VE systems.

Works like "ontology": world structure used by dialogue system in one of our examples.

Software engineering solutions

- UI event / update architecture
  - *Structuring method*
  - *Easy support for database tables (sets of tuples)*
  - *Publish/subscribe facility*
  - *Dynamical creation and deletion of UI components*
- Distributed software model

  network communication is transparent

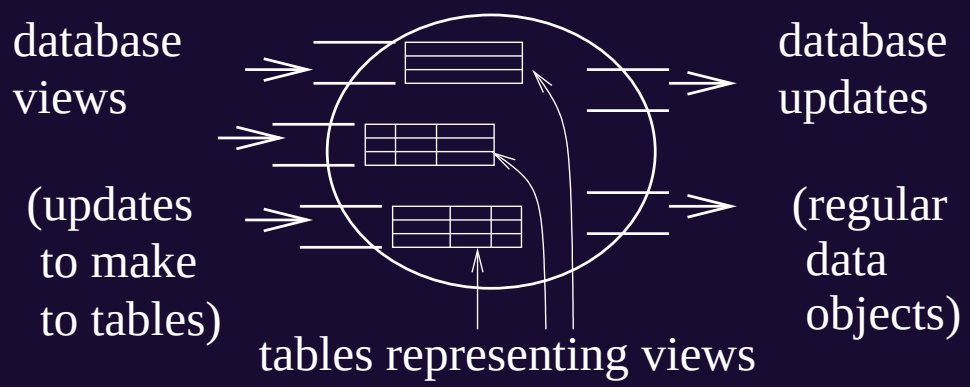
The VETk toolkit

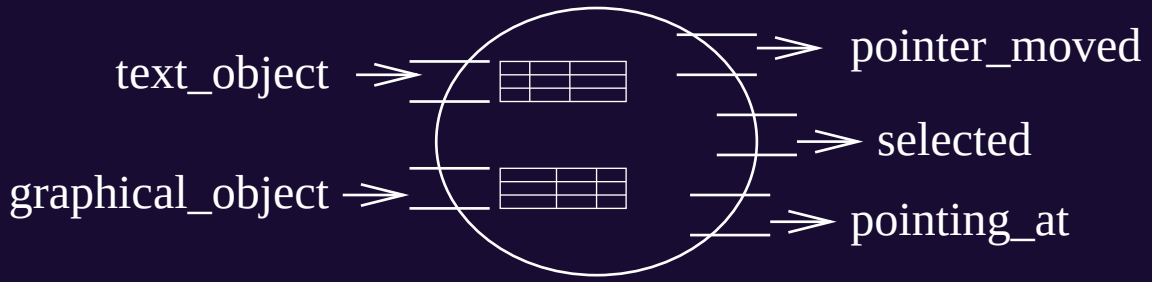
Software component is called **agent**. Each agent runs in **parallel** and has a number of **inputs** and **outputs**.

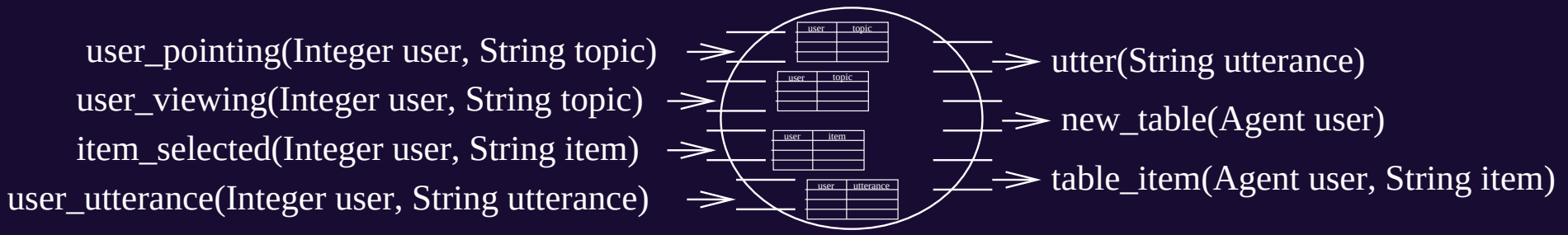
Each agent is **"glued"** into system by "glue" scripts, enabling **separation** of software components and system architecture.

The glue scripts contain **translation** of inputs and outputs to the **shared database**.

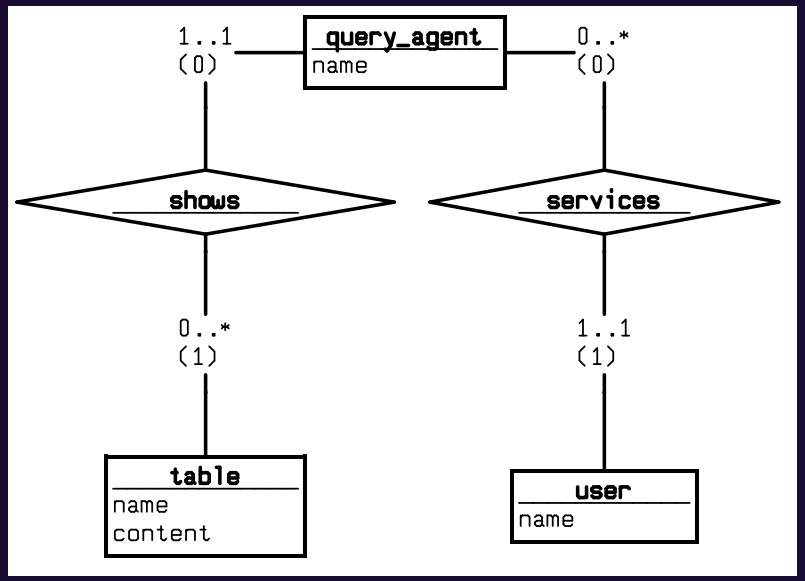
Toolkit consists of **multiple languages**, each for its own purpose.

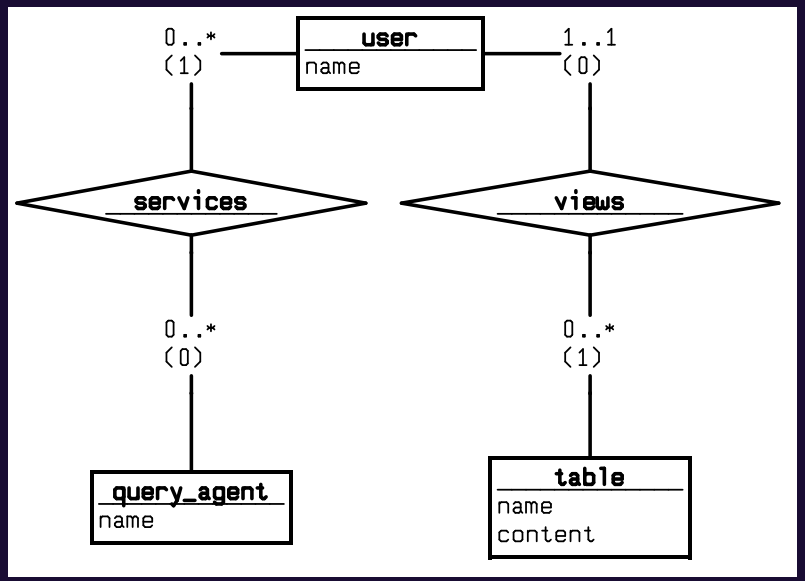

database
views
(updates
to make
to tables)
database
updates
(regular
data
objects)
tables representing views

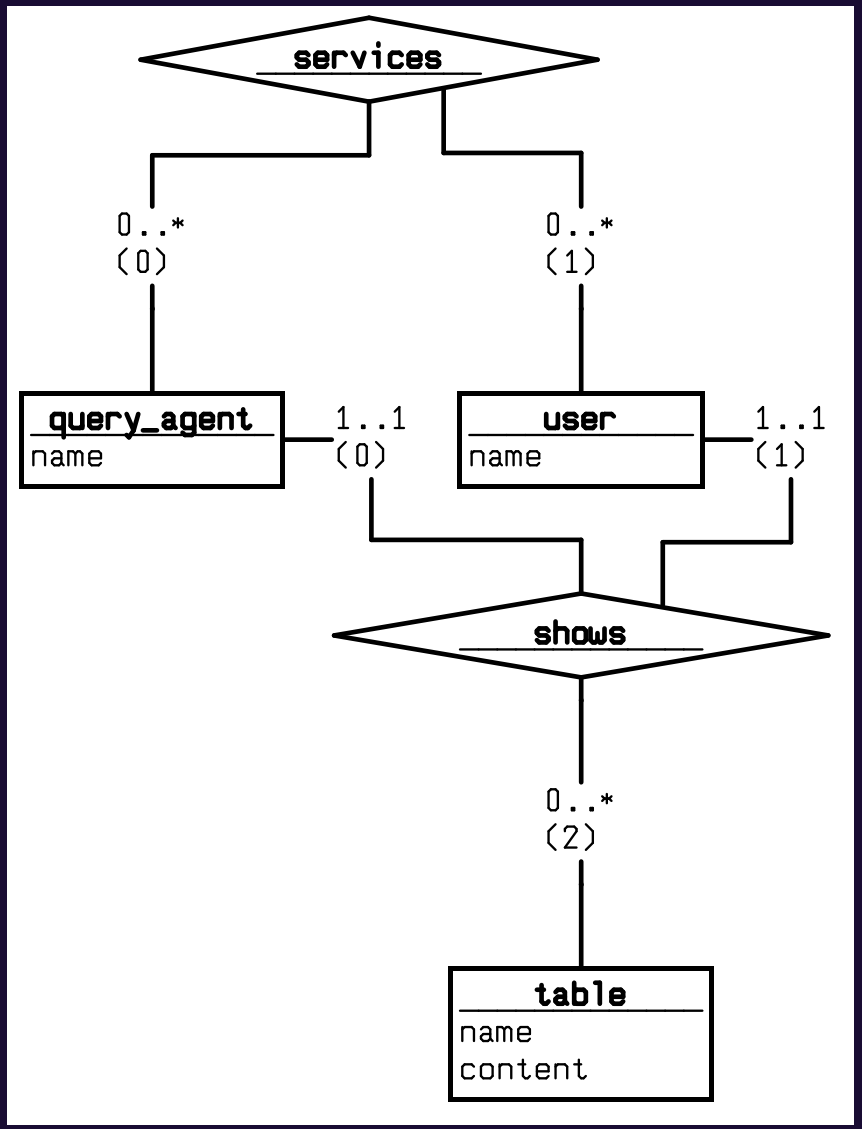

text_object
graphical_object
pointer_moved
selected
pointing_at

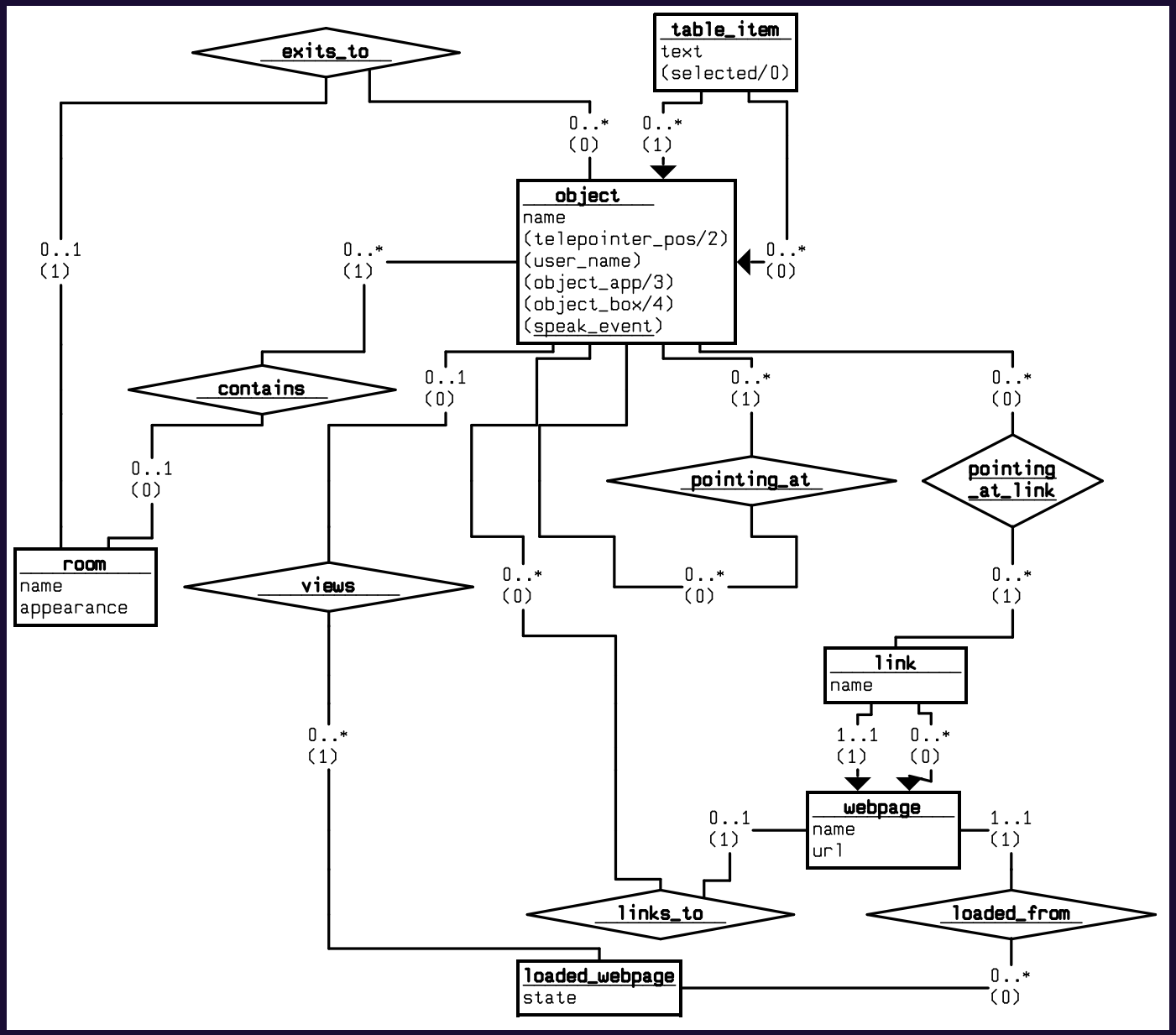

user_pointing(Integer user, String topic)
user_viewing(Integer user, String topic)
item_selected(Integer user, String item)
user_utterance(Integer user, String utterance)
user
topic
user
topic
user
item
user
utterance
utter(String utterance)
new_table(Agent user)
table_item(Agent user, String item)

System property constraints		
ERD of user interface	VDC (VETk Data Constraints)	non-expert
State automata of UI objects	VDC	non-expert
logic constraints on any data in database	VDC	expert
Glue languages		
Visual layout	HTML (with embedded VETkScript)	non-expert
Main glue language	VETkScript	expert
Individual agent behaviour		
Agents' interface	VCL (VETk Component Language)	expert
Agent behaviour	Java language and environment	expert


1..1
(0)
query_agent
name
0..*
(0)
shows
services
0..*
(1)
1..1
(1)
table
name
content
user
name


0..*
(1)
user
name
1..1
(0)
services
views
0..*
(0)
0..*
(1)
query_agent
name
table
name
content


services
0..*
(0)
0..*
(1)
query_agent
name
1..1
(0)
user
name
1..1
(1)
shows
0..*
(2)
table
name
content


table_item
text
(selected/0)
exits_to
0..*
(0)
0..*
(1)
object
name
(telepointer_pos/2)
(user_name)
(object_app/3)
(object_box/4)
(speak_event)
0..1
(1)
0..*
(1)
0..*
(0)
contains
0..1
(0)
0..*
(1)
0..*
(0)
0..1
(0)
pointing_at
pointing
_at_link
room
name
appearance
views
0..*
(0)
0..*
(0)
0..*
(1)
link
name
1..1
(1)
0..*
(0)
0..*
(1)
webpage
name
url
0..1
(1)
1..1
(1)
links_to
loaded_from
loaded_webpage
state
0..*
(0)

Conclusions

- It is possible to build a complex system
- Specifications remain short

$\Rightarrow$ Succesful as a proof of concept

Possible improvements:

- Error display, programming facilities, and language features
- Integration of 3D engine underway

For more information

see the VETk home page

http://wwwhome.cs.utwente.nl/~schooten/vetk/