CUIML: A Language For the Generation of Multimodal Human-Computer Interfaces

Christian Sandor
sandor@cs.tum.edu

Technische Universität München
Chair for Applied Software Engineering
Abstract

- **DWARF Project at the Technische Universität München**
- **UIML complied with some of our requirements**
- **Extension of UIML was developed and will now be presented**
What is DWARF?

- Distributed Wearable Augmented Reality Framework
- DWARF movie

Augmented Reality

Wearable Computer
The Problem

Framework for Presentation Layer of Wearable Computers

- An application should be platform independent
The Problem

Framework for Presentation Layer of Wearable Computers

- An application should be platform independent
- Interaction with the user is multimodal

Voice Input -> trigger -> Actions

Mouse Input -> Actions
The Problem

Framework for Presentation Layer of Wearable Computers

- An application should be platform independent
- Interaction with the user is multimodal
CUIML – a solution

- Cooperative User Interfaces Markup Language
- Based on UIML
  - Separation of document structure and presentation
  - Transformation to markup languages
  - Display in browsers for VRML, VoiceXML, HTML (Views)

- New concepts:
  - Controller
  - Manipulators
**UIML:**

- Addresses requirement of platform independence
- No support for multiple cooperating Views

---

**UIML document**

**Generic Structure**

```
A
  ↓
 B
```

```
C
  ↓
 D
  ↓
 E
```

**Mapping**

**HTML**

- A = H1
- B = H2
- C = H3
- D = H2
- E = H2

**WML**

- A = W1
- B = W2
- C = W2
- D = W7
- E = W7

---

**Generation Process**

1. **HTML**
   - H1
   - H2
   - H3

2. **WML**
   - W1
   - W2
   - W7
   - W7
Multimodal Human-Computer Interfaces consist of multiple Views

<table>
<thead>
<tr>
<th>Generic Structure</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
</tr>
<tr>
<td>B</td>
</tr>
<tr>
<td>C</td>
</tr>
<tr>
<td>D</td>
</tr>
<tr>
<td>E</td>
</tr>
<tr>
<td>F</td>
</tr>
<tr>
<td>G</td>
</tr>
<tr>
<td>H</td>
</tr>
<tr>
<td>I</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Mapping</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. VRML</td>
</tr>
<tr>
<td>B = 1</td>
</tr>
<tr>
<td>E = 2</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>2. VRML</td>
</tr>
<tr>
<td>C = 1</td>
</tr>
<tr>
<td>F = 2</td>
</tr>
<tr>
<td>G = 3</td>
</tr>
</tbody>
</table>

**Generation Process**

1. VRML  HTML  VoiceXML

**HCI 1**

<table>
<thead>
<tr>
<th>VRML</th>
<th>HTML</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
</tr>
</tbody>
</table>

**VoiceXML**

6

7 8

**HCI 2**

<table>
<thead>
<tr>
<th>VRML</th>
<th>WML</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>7</td>
<td>8</td>
</tr>
</tbody>
</table>
A *central component is needed*

- Describes state of the HCI
- Synchronizes views
A central component is needed

- Describes state of the HCI
- Synchronizes views

Deterministic Finite Automaton with XML based Configuration
CUIML extended with Controller

CUIML document

Controller Config =

Generic Structure

Mapping

<table>
<thead>
<tr>
<th>1. VRML</th>
<th>HTML</th>
<th>VoiceXML</th>
</tr>
</thead>
<tbody>
<tr>
<td>B = 1</td>
<td>C = 3</td>
<td>D = 6</td>
</tr>
<tr>
<td>E = 2</td>
<td>F = 4</td>
<td>H = 7</td>
</tr>
<tr>
<td></td>
<td>G = 5</td>
<td>I = 8</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>2. VRML</th>
<th>WML</th>
</tr>
</thead>
<tbody>
<tr>
<td>C = 1</td>
<td>D = 4</td>
</tr>
<tr>
<td>F = 2</td>
<td>H = 5</td>
</tr>
<tr>
<td>G = 3</td>
<td>I = 6</td>
</tr>
</tbody>
</table>

Generation Process

1. VRML

2. HTML

Controller Config =

HCI 1

VRML

Controller

VoiceXML

HCI 2

VRML

Controller

WML
Controller Example

State transitions of a View

VRML View

Controller
Controller Example

State transitions of a View

Start Print JOB?

VRML View

Controller
Controller Example

State transitions of a View

Print Job Started

VRML View

Controller
Views have to be modified by the Controller

- Access mechanisms for Views differ
Views have to be modified by the Controller

- Access mechanisms for Views differ
- Every View needs an appropriate Manipulator
- Described in behaviour section
Example of a Manipulator
Benefits of CUIML

- **Eases the task of developing multimodal HCIs**
- **Dynamic reconfiguration of the HCI to preserve functionality**
- **Incorporates the advantages of UIML**
  - Separation of development tasks
  - Platform independence
Future Work

- Rewrite renderers with JavaCC
- Adapt the Human-Computer Interface at runtime
Future Work

- Rewrite renderers with JavaCC
- Adapt the Human-Computer Interface at runtime
- Establish standard metaphors

http://www.cg.tuwien.ac.at/research/vr/pip/
http://www.csl.sony.co.jp/person/rekimoto/cube/