Collaboration Bus

Research Project Midterm Presentation

May 2005
Nicolai Marquardt
<firstname>.<lastname>@medien.uni-weimar.de
Cooperative Media Lab
Bauhaus University Weimar
Outline

1. Introduction and Related Work
2. Concept and Architecture
3. Technology, Implementation and Design
4. Example Scenarios
5. Summary
Motivation

• *Initial position: Existing infrastructure with software/hardware sensors and actors (at home and/or in the office)*

• Provide an easy-to-use method to integrate sensing and automated activities into everyday life (personal as well as business)

• Hide details of the technology (sensors, actors, platform) and the configuration of these components

• Consider personalized properties to represent the individual preferences and interpretations

• Exchange and share knowledge (and these personal interpretations)

• Control of the sensor-actor connections, but with guided support of the system
Introduction and Related Work

Motivation

- HW Sensors
- SW Sensors
- Locations
- User

Rules and Conditions

- Executes
- Away
- Occurrence
- Threshold
- Compared
- Available
- Working

Actors

Bauhaus University Weimar
Introduction and Related Work

Related Work (1)

1. Visual programming for UbiComp applications
   - Edit control networks for the in-home environment
   - Often with mobile device interfaces: PDA [Mavrommati et al. 2004], mobile phone [Barkhuus & Vallgarda 2003] or TabletPC [Humble et al. 2003]
   - Composition interface of sensors and conditions [Dey & Sohn 2003], Network editor [Mavrommati et al. 2004]
2. Sensor network composition software
   – Control over fine granular sensor network communication and processing [Baldwin et al. 2004]
   – Complete programming language, operator sets, math. libraries, etc.
   – Complex development environments, e.g. iConnect Micro-Epsilon
   – Integrated in business applications or science applications, e.g. Matlab

3. Visual programming languages
   – Enable the end-user to develop software components [Zhang et al. 2004]
   – Commercial products, e.g. HP VEE
Related Work (3)

4. Programming by demonstration
   - In Ubiquitous Computing: longer period of observation, then definition and learning phase, so that AI algorithms detect patterns [Dey et al. 2004]
   - Hide all specific details of the underlying mechanisms

6. Physical Representation
   - E.g. the Phidgets project of Saul Greenberg: associations between sensors, software and real objects [Greenberg & Fitchett 2001]
   - Implementation: ActiveX Controls in .NET
   - Also: ambient interfaces

[Dey et al. 2004]

[Greenberg & Boyle 2002]
Related Work

Findings: [Hague et al. 2001], [Cotterell & Vahid 2005], [Humble et al. 2003], [Cotterell et al. 2004], [Rodden et al. 2004], [Crabtree et al. 2004]:

- Users would like to control and program the sensor-based applications and build new ones; but without too much specific knowledge needed.
- They also would like to have the option, to change the behavior of their developed applications (dynamic) and to control them (activate, deactivate).
- Boolean conditional tables (logic blocks) are difficult to understand for the most users, so they need support.
- Users need support for using the interface (guide the user, hide optional parameters, use icons and color, etc.).
- Most of the conditional networks in the related work are built with three up to ten components and can be mapped to parallel pipes.
Related Work: Conclusions

The different approach of Collaboration Bus:

1. Focus especially on smaller data flow definitions to support and enhance work and everyday life
2. Intuitive, easy-to-learn interface, with specialized functions (not too generic)
3. Hiding the graph theory as much as possible
4. Sharing mechanism, exchange of data flows
5. The „repository“ view, personal control
6. Templates and patterns of flow definitions
7. No replacement for the complex „inference engine“ modules
8. Covers *Ubiquitous Computing* and CSCW applications
Building Sensor-based Applications

1. Hardware level: nodes communication
2. Using framework: adapters, classes, hard coded
3. Widgets for software development (e.g. Context Toolkit)
4. Software development using libraries, visual IDE (e.g. Phidgets with ActiveX controls)
5. Visual editor, editing data flow networks, graph theory
6. Visual editor, guided network composition, adaptive
7. Wizard dialog, step-by-step pages
8. Automatic application (network) creation based on problem description, AI algorithms

User centric
Sens-ation: New Applications

Concept and Architecture
Collaboration Bus

Personal Repository
- SensWidget A
- SensWidget B
- SensWidget C

Configure, Components

Create, Edit, Upload SensWidget Description

Create, Instance, Parse, XML

Overview, State, etc.

Activate, Deactivate, Share, etc.

Editor

Control

Sensation Platform

S1

S2

A1

A2

A3
Sharing and Templates (1)

1. Share SensWidget Events

SSW A

SSW B

2. Share SensWidget as template

SensWidget A

SensWidget D

3. Share Complete SensWidget (adaptive)

SensWidget B

SensWidget E

SensWidget C
Sharing and Templates (2)

1. Reuse SensWidget Events as source for other widgets

2. Reuse pipe data flow as template

3. Reuse the complete SensWidget as template (copy)
Layers

Structured into separate layers:

- Presentation layer: editor and personal repository
- Data exchange: description of current data flows and parameters
- Instance layer: creating components of the inference processing
- Personal Repository: individual SensWidgets, running threads
- Global repository: sharing, components, wrappers, etc.
Concept Overview

Key features of “Collaboration Bus”:

1. *Personalized*: Instead of using only a few predefined and generic operational networks we use quite a number of **highly personalized sensor-actor-relations**

2. *Overview and control*: Let the user control (start, stop) and edit his own created relations

3. *Easy and fast*: these sensor-actor-relations have to be defined and executed in a few minutes (seconds)

4. *Sharing knowledge*: for cooperative work, users can share their sensor-actor-relations (adaptive modules)
Implementation

Terms and definitions:

- **Task (Procedure, Workflow):** This is the activity the user wants to implement with his tools (infrastructure)
- **Data Flow:** the stream of data between source and sink, passing several processing steps
- **Pipe:** connection between two components
- **Component:** source, sink and filter
- **Composition (network, graph):** directed graph of connections between data sources, sinks and filters, and with specification of their relations
- **Filter (operator):** conditions, operations, calculations, data processing, enrichment with meta data
- **SensWidget:** encapsulated composition with additional features, especially for sensor-actor-relations
Implementation

Project Parts (Sub Projects):

1. **Instantiation**: The composites of sources, filters, operators and sinks, local or remote objects, dynamic creation (from XML)  
   Complexity: ++++

2. **Composition**: User interface, editor, support for creating new SensWidgets  
   Complexity: ++++

3. **Control**: Personal repository, updates, start editor, summary, instantiate modules  
   Complexity: ++

4. **Actors**: Actor elements, display, hardware, XML description, handling in Sens-ation server, ActorBridge  
   Complexity: +++

5. **Language**: Description language for sensor-actor-data-flow, networks, operators, parameter  
   Complexity: +++

6. **Further Sens-ation Extensions**: Extensions of the sensor XML description, actor integration, metadata, SensorValue extensions (pipes), personalization, *user right management, security*  
   Complexity: +++
Technology

• Filter and pipe design pattern: processing components as filters, thread-safe pipes (FIFO) for data transfer, most of the components multi-threaded
• Abstract classes and interfaces: actor, source, pipe, filter
• Dividing of smaller processing steps into command objects; reusable command collections
• Composites (GoF) of components: filter with hierarchical structure
• Each component with an GUI representation: display method and also for dynamic set of parameters (reflection)
• Boolean operators for AND and OR conditions between pipelines
• Transformation between the main data types (int, boolean, String)
• Two implementations of filters and operators:
  – Multi-threaded objects for independent processing nodes
  – Entries in the command set collection (composite), especially for pre-processing of the values before they enter the pipes network (e.g. normalize, threshold, cut)
Filters and Operators

• Processing node of the pipeline (implement source and sink interface)

• Adaptive; include sensor parameters (e.g. normalize)

• Operator types (nodes):
  – **Numeric**: e.g. threshold, gate, counter, average, normalize, cut
  – **Binary**: e.g. gate, convert
  – **Text**: e.g. match, match list, convert, length, regular expressions
  – **XML**: e.g. contains node, compare, parameters, text search, other string filters, count entries, root object, match DTD
  – **General**: e.g. event counter, collector (reports), time gate, multiple translations between the data types, database lookup, sensor, delay

• Message generator for reports or summary (includes sensor and context details)
Actors

- Set of hardware and software actor modules (connected via adapters)
- Actor collections: ActorBridge
- XML description for registration:
  - 1 x Callback connection, location, availability, other parameters
  - N x Command + Parameter: executes action of the actor
- Examples of the actors set:
  - Software: Email, RSS feed, application starter (e.g. AppleScript Lib), visual notify (e.g. dockbar), IM daemon sends message
  - Hardware: controllable power plugs, IR commands, acoustic notification, SMS message
- Types of messages: event, report, summary, collection
Interface Design Studies

1. Control Interface:

- Overview of created sensor-actor-relations (SensWidgets)
- SensWidget operations: start, stop, edit, delete, add
- See current state of all Widgets
- Start and stop complete repository
- Additions: sharing options panel, external flow definitions
Interface Design Studies

2. Editor Interface:

- Create a new SensWidget
- Upper screen: name, description, configuration, repeating, templates and sharing options
- Create data flow conditions: select sensor, configure sensor, add conditions and finally select actor
Interface Design Studies
Using SensWidgets

- Repeated actions, loops (e.g. weekly report, awareness, home environment control)
  - More complex compositions
  - Also use of multiple pipes and inference engine modules

- Only once and reject (e.g. notification of an awaited event, contact a person)
  - Short pipelines
  - Often using templates and copy mechanism
  - Specify max. execution time

- Multiple time execution, manual select (e.g. project messages, state)
  - Specify execution conditions (e.g. with filter element)
Using SensWidgets

• Ubiquitous Computing, private usage:
  – Control the home environment
  – Awareness of peoples activities
  – Device associations
  – Remote control
  – Reminders

• Computer Supported Cooperative Work, business usage:
  – Contact colleagues (with the option to select the most convenient point of time)
  – Project reports (from various sources)
  – Awareness of project participants
Example Scenarios: UbiComp

- **Movement**
  - Threshold = 0
  - State: night

- **Movement home**

- **Light sensor**
  - Time gate: 8 a.m. < x < 6 p.m.
  - System: Office

- **Personal Login**
  - Random value, Every 30 min

- **Time counter 30 min**

- **Lights off**

- **Dockbar color matcher**

- **Power Socket Control**
  - Time mapping
  - Time gate: 6 p.m. < x < 11 p.m.

**Processing and Conditions**

**Actors**
Example Scenarios: CSCW (1)

- **Eclipse CVS**
  - Keyword: Project name
  - Counter: > 5
  - System: Office
  - Time gate: > 6 p.m.

- **Personal Login**
  - Personal Login System: Office
  - Time gate: > 6 p.m.

- **Login**
  - User list: Tareg, Nic
  - Application Group: Development

- **OS Sensor**
  - Application Group: Development

- **BSCW**
  - Keyword: Project name
  - Counter: > 10
  - Threshold: high

- **Movement Lab**
  - Threshold: high

- **Send Email report**
  - Send SMS notification

- **Start iChat + invitation**

← Processing and Conditions →

→ Actors →
Example Scenarios: CSCW (2)

Encapsulation as inference engine

Login
- User list: TG
- System: Office

Personal Login
- System: Office
- Time gate: 7 a.m. < x < 8 p.m.

OS Sensor
- Application match: PowerPoint
- Application state: not presentation mode

Phone Sensor
- State: not active
- Delay: 1 minute

Movement
- Threshold: > 1

Acoustic Notification

Visual feedback: Color light

SensorSink: Nic available + share

← Processing and Conditions
Actors →
Conclusion

• User interface and control of light-weight SensWidgets
• Easy-to-use editor, reusable components, templates
• Sharing for collaboration
• Flexible underlying technology layer: XML, dynamic component composition, transferable
• Personal repository, overview and control (local or remote located)
• Concept for the integration of Actors, and some sensor description extensions at the SensBase platform
**Literature and References (1)**

[Barkhuus & Vallgarda 2003] Luise Barkhuus and Anna Vallgarda: Smart Home in Your Pocket, in: Interactive Posters, Interfaces, Adjunct Proceedings of the UbiComp, Seattle, USA, 2003


<table>
<thead>
<tr>
<th>Reference</th>
<th>Title</th>
</tr>
</thead>
</table>
Thank you for your attention!

Nicolai Marquardt
Collaboration Bus Research Project
Cooperative Media Lab
Bauhaus University Weimar