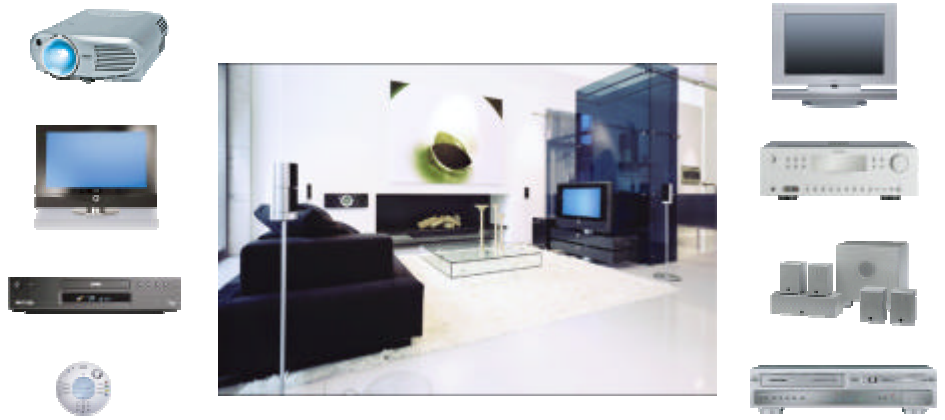
Distributed Implementation of a Self-Organizing Appliance Middleware

sOc-EUSAI 2005
Conference of Smart Objects & Ambient Intelligence
October 12th-14th 2005
Grenoble, France
Oral Session 6 - Middleware

Michael Hellenschmidt
Fraunhofer-Institute
for Computer Graphics
Fraunhoferstrasse 5
64283 Darmstadt
Germany

Application Domain: Living room



Heterogeneous Devices (from different vendors)…

Application Domain

… form an ad-hoc device ensemble.

Application Domain: Meeting room







Spontaneous Meeting in an „empty“ room…

Application Domain: Meeting room

… devices form a cooperative meeting room.

Structure

- Requirements of a self-organizing middleware
- A view on state-of-the-art-models for device ensembles
- The SodaPop-Modell and its distributed implementation
- Summary and Discussion

- Only ideas and approaches, no technical details !!!

Requirements for Self-Organization

- ***Devices*** should be ***independent*** and should be able to act ***stand-alone***
- Devices should be ***exchangeable***
- There must be ***no central device*** (because a central controler is a contradiction itself to the demand of ad-hoc self-organization)
- ***Distributed implementation*** should be supported (to guarantee decentralized communication)
- Mechanisms that ***guarantee the data-flow*** between the devices, that means effective ***service discovery resp. conflict resolution mechanisms***

IGD
••• Fraunhofer Institute for Computer Graphics ••• Department In teractive Multimedia Appliances •••

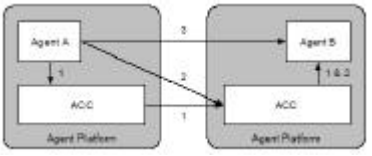
Middleware Assessment

sOc EuSAI 2005

Point-to-point messaging technologies:

- White page service
- Yellow page service (facilitator)
- Publish/subscribe mechanisms
- Service discovery / service composition mechanisms are internal functions of each component

Examples: KQML / FIPA solutions



KQML (Knowledge Query and Manipulation Language):
Tim Finin, Richard Fritzson, Don McKay, and Robin McEntire,
KQML as an Agent Communication Language,
In: *Proc. of the 3rd Int. Conf. on Information and Knowledge Management* (CIKM'94),
p. 456-463, ACM Press, 1994

FIPA (Foundation for Intelligent Physical Agents):
Jeremy Pitt and AbeMamdani,
Some Remarks on the Semantics ofFIPA's Agent Communication Language,
In: *Autonomous Agents and Multi-Agent Systems*,
2(4):333–356, 1999

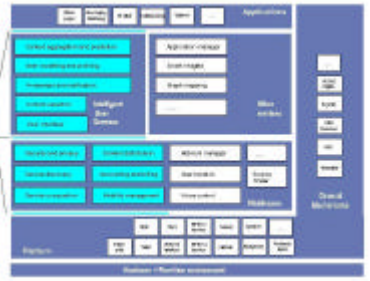
Fraunhofer Institut Graphische Datenverarbeitung
Technische Universität Darmstadt

Middleware Assessment

Some Notes:

- ***Hard-wired*** communication flow (Agent X → Agent Y)
- Publish/subscribe-mechanisms give information to any interested component → ***possible conflicts during further processing stages***
- The term component and its possible functions (resp. the used ontology) is not well defined.
- Thus: Internal service discovery mechanisms
  - each developer / programmer is responsible for implementing appropriate strategies !
  - Predictability of overall system behavior ?
- No ***real cooperative self-organization*** of device ensembles

Examples: Architecture of AMIGO device



AMIGO short project description, January 2004):
available from: http://www.amigo-project.org

Amigo D2.1: Specification of the abstract system architecture
Nikolas Georgantas (Eds.), April 2005
Available from: http://www.amigo-project.org

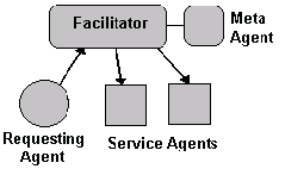
Middleware Assessment

Middleware technologies with ***specific components that control the communication*** flow, like:

- Routing components
- Meta agents
- Evaluation agents
- Central registries

These components are using ***rules*** to avoid potential conflicts between competing components.

Open Agent Architecture



Open Agent Architecture:
Martin, D.L., Cheyer, A.L., and Moran, D.B., *The Open Agent Architecture: A Framework for Building Distributed Software Systems*, in: *Applied Artificial Intelligence*, Vol. 13, No. 1-2, pp. 91-128, Jan-Mar 1999

Galaxy Communicator:
Seneff , S., Lau, R., andPolifroni , J., *Organization, Communication, and Control in the Galaxy-II Conversational System*, in: *Proc. of Eurospeech'99*, pp. 1271–1274, 1999

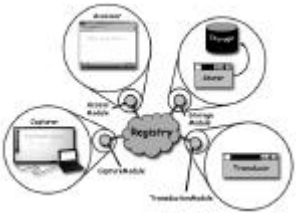
Galaxy Communicator

Middleware Assessment

Some Notes:

- Routing ***rules have to be changed*** when device ensemble changes (devices leave or enter the ensemle)
- Devices / Components that host the "rules/evaluators" ***must exist*** resp. are not allowed to leave the ensemble
- ***No ad-hoc cooperation*** / networking

INCA-Infrastructure



INCA:
Truong, K.N., andAbowd G.D. (2004) *INCA: A Software Infrastructure to Facilitate the Construction and Evolution of Ubiquitous Capture and Access Applications*, in: *Proc. of the 2nd Intern. Conf. on Pervasive Computing (Pervasive 2004)*

Jaspis :
Turunen, M., Hakulinen, J., Räihä, K-J., Salonen E.- P., Kainulainen , A., and Prusi(P. (2005) *An architecture and applications for speech-based accessibility systems* , in: *IBM Systems Journal*, Vol. 44, No. 3, 2005

Jaspis -Framework

Again the requirements…

Updated ***requirements concerning ad-hoc networking***:

- We want to support DEVICE ensembles
- Add-hoc cooperation of heterogeneous DEVICES (not components)
- Note: Users are not interested in components, they want that their devices run

Consequently:

- ***Component is a virtual entity*** that runs on a device
- ***Device is a physical entity***. It can be the host of several components

- For ad-hoc networking of components we must ensure ***ad-hoc networking of devices***. Devices are the smallest granularity that should be supported.

Again the requirements…

Updated ***requirements concerning effective Service Discovery*** resp. Conflict Resolution Mechanisms:

- Those mechanisms have to be as ***transparent*** as possible to ***guarantee reliability*** of the entire device ensemble
- The implementation of those mechanisms ***should not increase the effort*** of the software engineers

Consequently:

- There must be a the possibility to ***define a fixed set of possible kinds of components*** (note: not a fixed set of components)
- Then: ***Predefined and usable service discovery*** resp. conflict resolution mechanisms to control the message flow can be provided (for the software engineer)

Middleware model SodaPop

Some SodaPop facts:

- Self-Organizing Data-flow Architectures suPporting Ontology-based problem decomPosition)
- Basic Elements: Components and Component Groups
  - ***Components / Transducers:***
    - Working „between“ ontologies, that means they map messages from one ontology to another
    - Sending messages (source of messages)
    - Competing against for messages (destination of messages)
  - ***Component Groups / Channels***
    - Responsible for delivering the messages
    - In generell ontology-dependent
    - Executing conflict resolution mechanisms / Service Discovery strategies
    - Set of groups can be defined (according to the application domain)

SodaPop and Topologies for Ambient Intelligence:

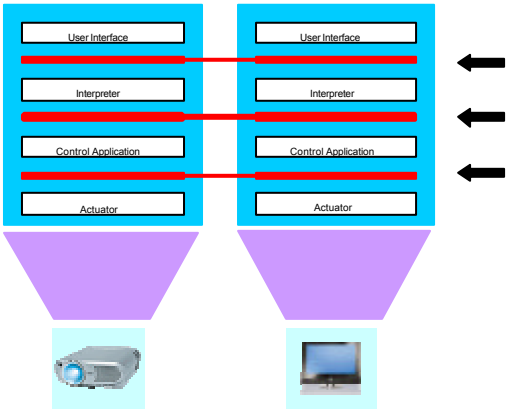
Heider T., Kirste T.,
Architecture consideration for interoperable multi-modal assistant systems,
In: *Proc. 9th Intern. Workshop on Design, Specification, and Verification of Interactive Systems (DSV-IS 2002)*, Rostock, Germany, 2002

Hellenschmidt M., Kirste T.,
SodaPop: A Software Infrastructure Supporting Self-Organization in Intelligent Environments,
In: *Proc. of the 2nd IEEE Conference on Industrial Informatics, INDIN 04*, Berlin, Germany, 24-26. June, 2004.

Hellenschmidt M., Kirste T.,
A Generic Topology for Ambient Intelligence,
In: *Proc. of the Second European Symposium on Ambient Intelligence (EUSAI) 2004*, Eindhoven, the Nederlands, November 8-10, 2004

Illustration

Two devices that share the same component groups:

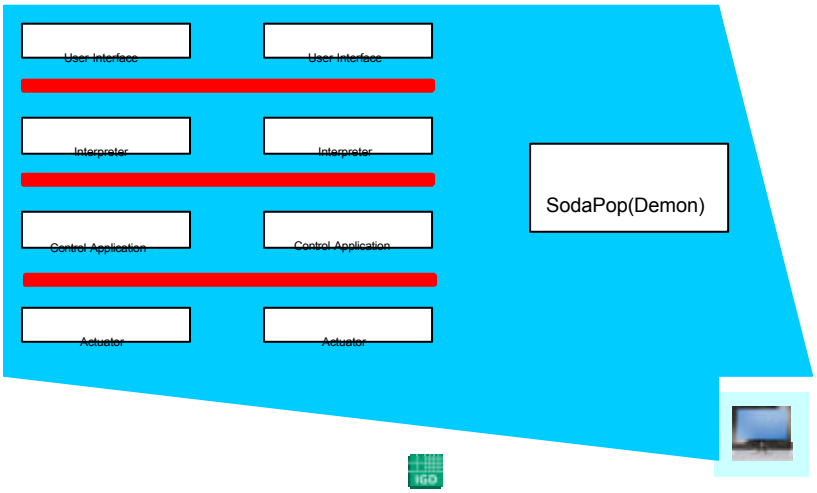


- Three Component-Groups that “discussing” about messages of the same semantic (ontology)
- Possibly three different conflict resolution resp. service discovery strategies

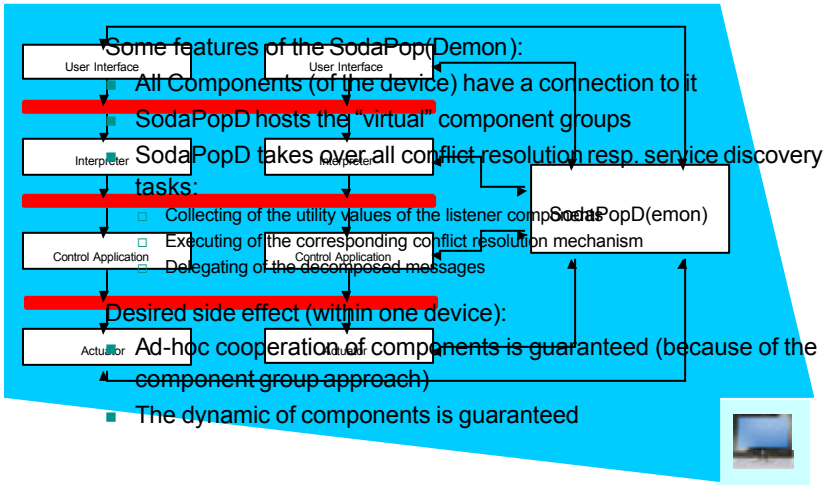
Hellenschmidt M., Kirste T.,
A Generic Topology for Ambient Intelligence,
In: *Proc. of the Second European Symposium on Ambient Intelligence (EUSAI) 2004,*
Eindhoven, the Nederlands, November 8- 10, 2004

Implementation

…each physical device has its own SodaPop-Service

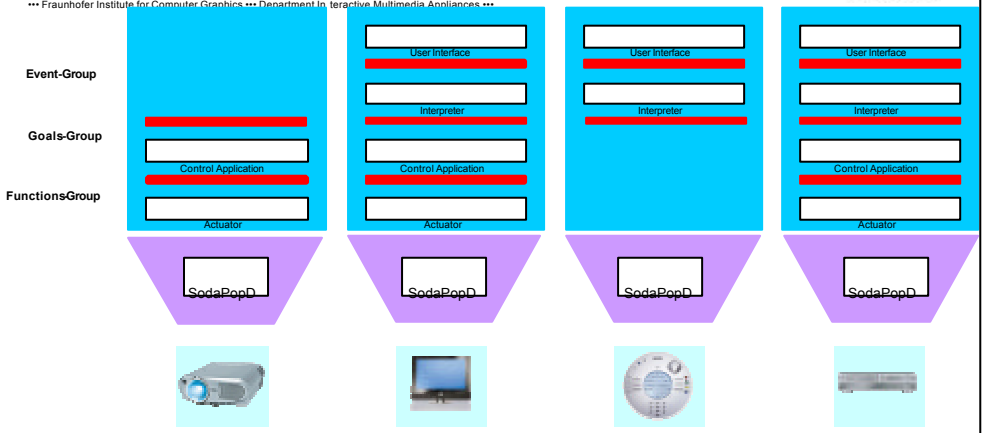


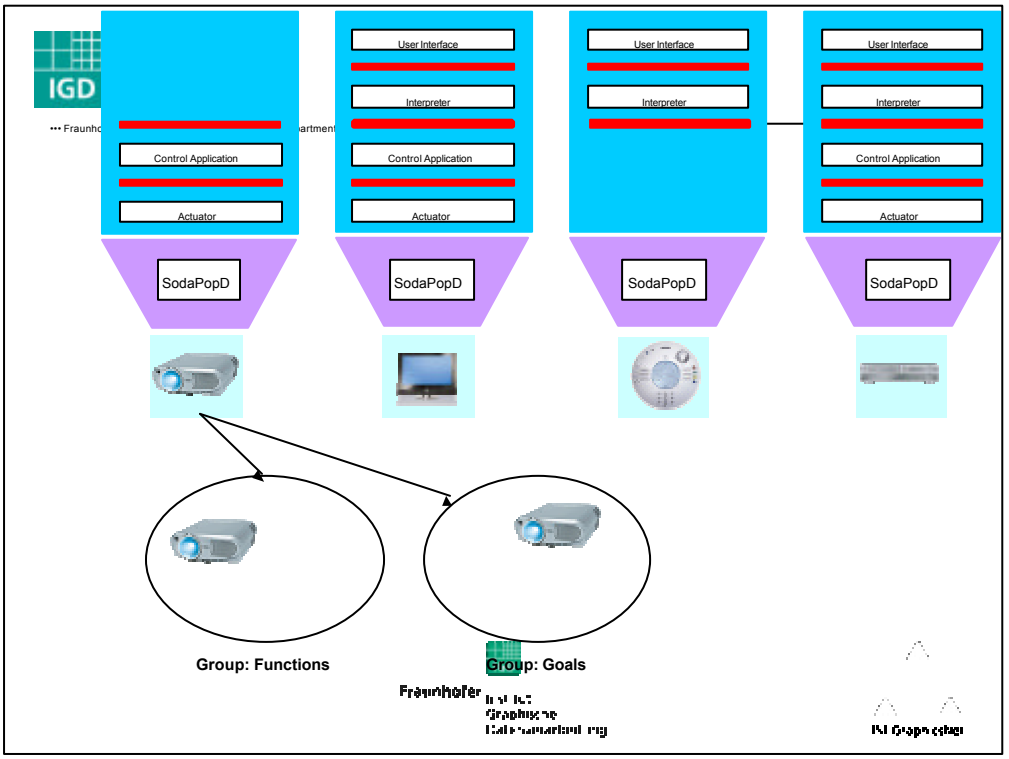
Implementation proposal

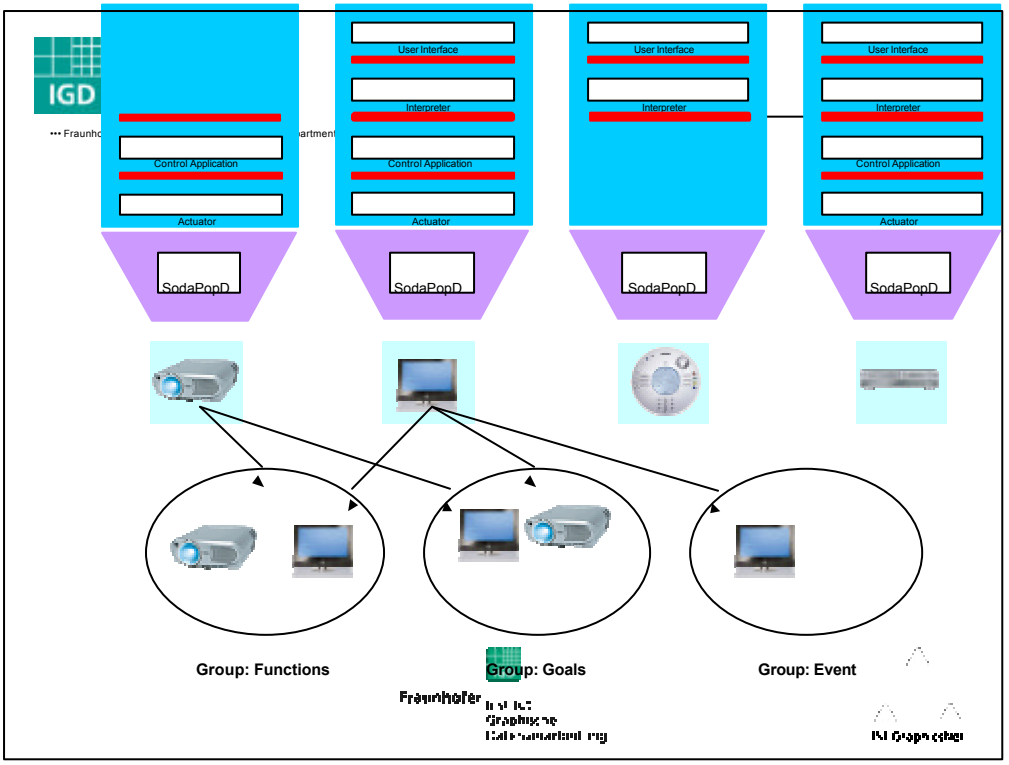


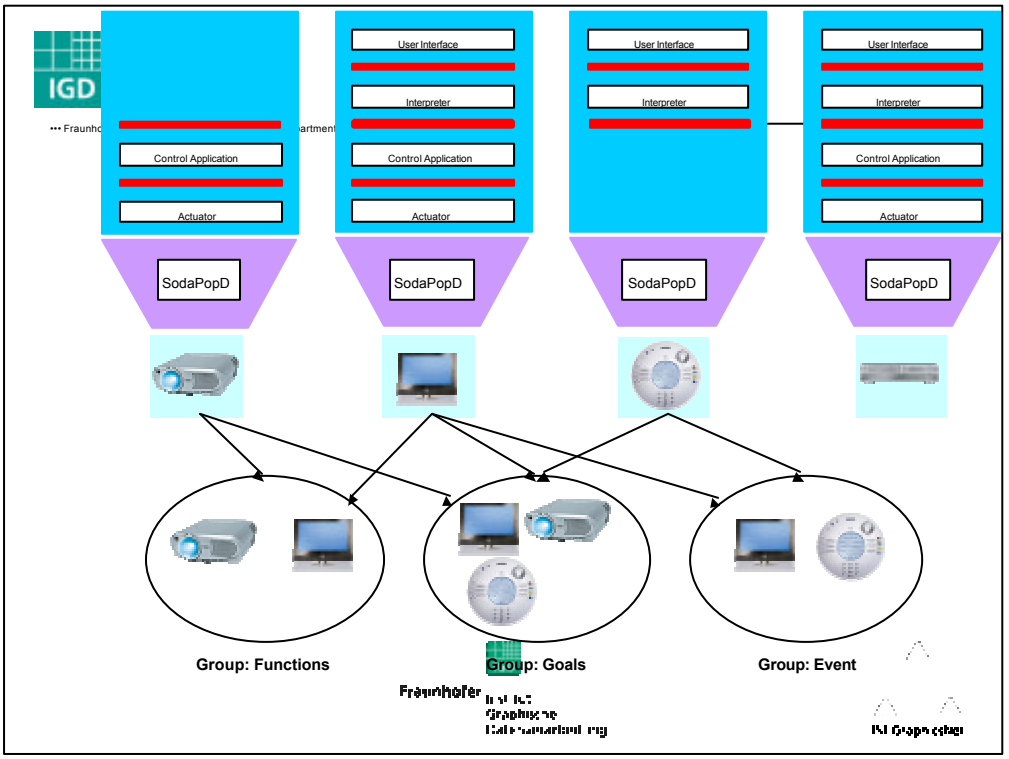
- Some features of the SodaPop(Demon):
  - All Components (of the device) have a connection to it
  - SodaPopD hosts the "virtual" component groups
  - SodaPopD takes over all conflict resolution resp. service discovery tasks:
    - Collecting of the utility values of the listener components
    - Executing of the corresponding conflict resolution mechanism
    - Delegating of the decomposed messages
- Desired side effect (within one device):
  - Ad-hoc cooperation of components is guaranteed (because of the component group approach)
  - The dynamic of components is guaranteed

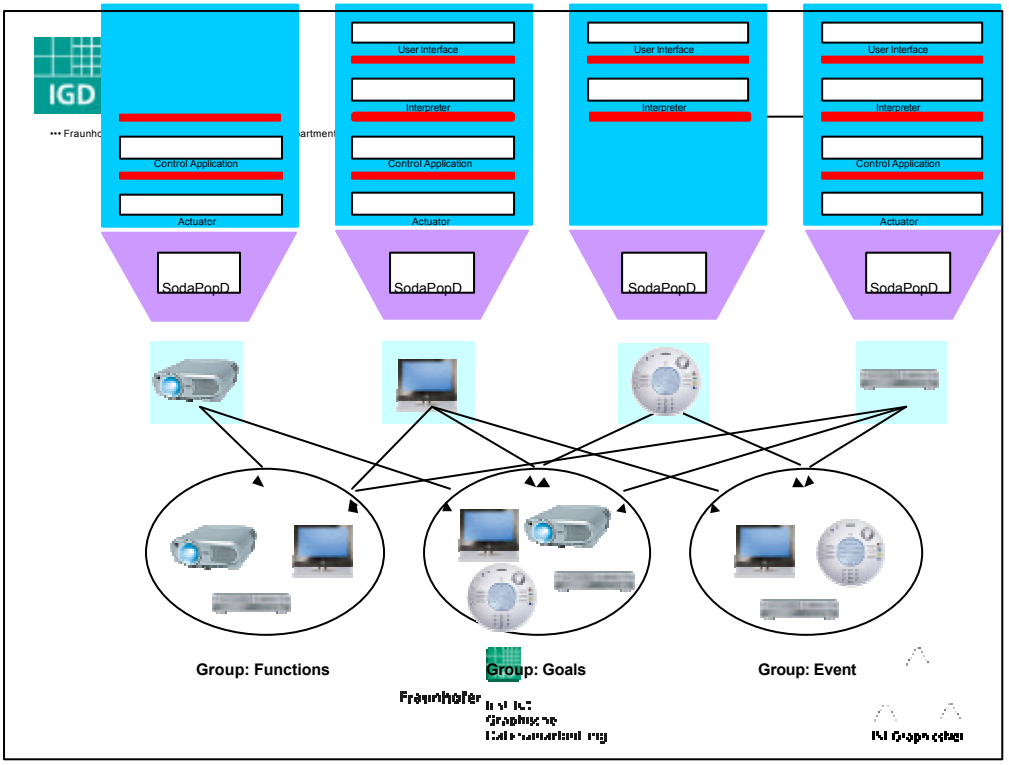
Advanced Example




IGD
User Interface
Interpreter
Control Application
Actuator
SodaPopD
Group: Functions
Group: Goals


IGD
User Interface
Interpreter
Control Application
Actuator
SodaPopD
Group: Functions
Group: Goals
Group: Event


IGD
User Interface
Interpreter
Control Application
Actuator
SodaPopD
Group: Functions
Group: Goals
Group: Event


IGD
User Interface
Interpreter
Control Application
Actuator
SodaPopD
Group: Functions
Group: Goals
Group: Event

Summary

Two novel ideas / approaches (with respect to other middleware concepts):

- Quite simple:
  Differentiation between ***virtual components and physical devices***
  → SodaPopD(emon)s as hosts on each device as representative of its device components
- Not simple:
  Making ***service discovery resp. conflict resolution mechanisms transparent*** and available (for software engineers) by:
  - Possibility to define a set of component-groups (with certain ontologies)
  - And then:
    possibility to identify possible conflict and to provide conflict resolution strategies for the different groups

IGD
••• Fraunhofer Institute for Computer Graphics ••• Department In teractive Multimedia Appliances •••
sOc EuSAI 2005

Summary

That means:

1. Define a common set of component groups (in dependence of the application domain and the used ontologies)
2. Identify possible service discovery conflicts and provide solutions for them
3. Make conflict resolution mechanisms for service discovery transparent and available within the SodaPopD(emon)

4. Implement some components that use the SodaPopD(emon) functions
5. Tie everything together

Fraunhofer Institut Graphische Datenverarbeitung
IGD Graphics

What is not mentioned here….

… but in the paper and the references:

- Peer-to-peer and broadcast communication processes of the SodaPopD(emon)s by using JXTA resp: UPnP as underlying technology
- Explicit conflict resolution strategies for the illustrated example
- The explicit execution of conflict resolution resp. service discovery mechanisms within a group of SodaPopD(emon)s

••• Fraunhofer Institute for Computer Graphics ••• Department In teractive Multimedia Appliances •••

Results:

sOc EuSAI 2005

- ***Real ad-hoc cooperation*** of physical devices (and its hosted components)
- ***Software engineers are disburdened***, because they only have to integrate their components into the pre-defined component-groups (of the application domain) and use automatically the appropriate service discovery resp. conflict resolution mechanisms
- First ***Demonstrators and Applications***, e.g. *Christian Elting "Orchestrating Output Devices – Planning Multimedia Presentations for Home Entertainment with Ambient Intelligence", Talk on sOc-EUSAI 2005*
- Some example downloadable from ***http://www.dynamite-project.org***

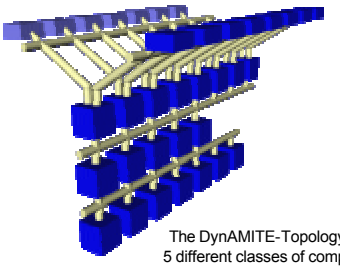
Fraunhofer Institut Graphische Datenverarbeitung

IMC Chemnitz

Thank you for your attention

DynAMITE

- Dynamic Adaptive Multimodal IT Ensembles
- Goal:
  - Develop a semantic middleware supporting context-aware self-organization of multimedia appliances
  - Enable the construction of coherent ensembles
  - Provide a public reference implementation
- Partners
  - EML, Fraunhofer-IGD (Lead), Loewe Opta
- Pointers:
  - www.dynamite-project.org
  - Some software downloadable



The DynAMITE-Topology differs 5 different classes of components

••• Fraunhofer Institute for Computer Graphics ••• Department In teractive Multimedia Appliances •••