O3MiSCID, a Middleware for Pervasive Environments

Rémi Emonet, Dominique Vaufreydaz, Patrick Reignier, Julien Letessier

Rémi Emonet
INRIA Rhône-Alpes
SIPE06, Lyon, France
29, June 2006
Motivations

- Stimulating synergy in/between research teams
  - Software reuse
  - Software interoperability
- Simplifying conception of distributed applications
  - Unified communication means
  - Programmatic facilities
- Enabling higher level concepts
Outline

- Requirements
- Middleware presentation
  - Abstract architecture
  - Detailed architecture
- Examples
- Conclusions
Middleware Requirements

- **Attractiveness and availability**
  - Available to everyone
  - Low threshold, high ceiling

- **Network efficiency**
  - High bandwidth
  - Low latency
Middleware Requirements

- Robustness
- Ease of configuration
  - Dependencies between services
- Extensibility
  - Context awareness
  - Autonomous computing
Outline

- Requirements
- **Middleware presentation**
  - Abstract architecture
  - Detailed architecture
- Examples
- Conclusions
O3MiSCID

- **Object-Oriented**
- **Opensource**
- **Middleware for**
- **Services Connection,**
- **Inspection and**
- **Discovery**
Abstract Layered Architecture

- **Layer 1: Network Communications**
  - Splitting network streams in messages
  - Arbitrary message content

- **Layer 2: Services**
  - Service declaration and advertisement
  - Service discovery and inspection

- **Layer 3: Ontology**
  - Semantic service description
C: C++ API
CJT: C++/Java/Tcl APIs

O3MiSCID L0 platform independence

O3MiSCID L1 network communications

O3MiSCID L2 services decl./discov.

O3MiSCID L3 ontology

DNS-SD
Outline

- Requirements
- **Middleware presentation**
  - Abstract architecture
  - Detailed architecture
- Examples
- Conclusions
L1: Network Communications

• Basic Interconnection Protocol (BIP)
  – Splits the communication stream into messages
  – 34-byte header
  – Allows arbitrary content type

• Programmatic facilities
  – Communication channel instantiation
  – Message interpretation: string, xml, binary
L2: Services - Declaration

- Black box software unit
- Communication channels
  - Input, output, bidirectional
- Variables
  - Access rights
  - Remote modifications, modification subscription
- Control channel
  - Inspection, variable related queries
Example Services
Example Service Declaration

```java
service = registry.register("Camera");
service.addConnector("videoOutput",
    "Frame by frame output", InOutputKind.Output);
service.addVariable("calibration", "Matrix4x4", "read");
...
```
Example Service Declaration (using XML)

```xml
<service name="Camera">
    <variable name="calibration">
        <access>read</access>
        <type>Matrix4x4</type>
    </variable>
    <output name="videoOutput">
        <description>Frame by frame output</description>
    </output>
</service>
```

```
service = registry.registerFromXML(xmlStream);
```
L2: Services - Discovery

- DNS-SD over multicast DNS
  - Distributed service advertisement
  - Quick connection/disconnection notifications
- Custom filters for service selection
  - Service name, hostname, owner
  - Channel presence, channel description
  - Variable value
Example Service Discovery

```
bipService.addConnectorListener("videoInput", this);

remoteProxy = bipService.findService(
    and( namels("Camera"), ownerIs("John")));

bipService.connectTo("videoInput", remoteProxy, "videoOutput");
```
L3: Ontology

- Semantic description of services
- Simple class hierarchy
  - Single inheritance
  - Single class belonging
  - Class fields
- Service lookup based on
  - Service class
  - Service fields values
Implementation

- Cross-language and cross-platform
  - Java, Osgi
  - Portable C++ (Windows, Linux, MacOSX)
  - Tcl

- APIs
  - Close to each other
  - Exploiting particularities of each language
  - User centered abstractions
Outline

● Requirements

● Middleware presentation
  – Abstract architecture
  – Detailed architecture

● Examples

● Conclusions
Example: Plotting Room Configuration

- Browse for interesting hardware services
- Query interesting parameters (position, orientation, ...)
- Render the room with its hardware
Example: Automatic Cameraman

- Automatic cameraman
  - Models the evolution of a presentation
  - Chooses the appropriate point of view
  - Acts as a movie director

- Communications
  - Use O3MiSCID
  - From audio and video stream to sparse eventing
Automatic Cameraman

Diagram showing the components and connections involving Camera, MovieDirector, Computer2, Computer3, ContextModeler, Tracker2D, SpeechActivit..., and Microphone.
Example: System Monitoring

- Browse for interesting services
- Inspect their interconnections
- Automatically generate a graph
System Monitoring
Outline

- Requirements
- Middleware presentation
  - Abstract architecture
  - Detailed architecture
- Examples
- Conclusions
Conclusions

- **Motivations**
  - Stimulate synergy in/between research teams
  - Simplify distributed application design

- **Layered architecture**
  - L1: Network communications
  - L2: Services declaration and discovery
  - L3: Semantic service description
Conclusions (Cont'd)

- Implementation
  - User centered design (ease of use, high availability)
  - Open source release (soon publicly available)
    - Source code, tutorials

- Status
  - Upper layer (L3) is work in progress, lower ones are stable
  - Good adoption rate, used in 4 research teams

- Good base for higher level works
Thank You For Your Attention
QUESTIONS?
Future “engineering” work

- Performance optimizations in particular cases
  - Shared memory use on local machine
  - Direct method calls between services running in a same Java Virtual Machine
  - Generalized proxy architecture

- Other implementations
  - .Net implementation (C#)
  - Python implementation (and JSR223?)
Generalized Proxy Architecture

Diagram showing the generalized proxy architecture with interactions between producers and consumers in a networked environment.
Why a custom solution?

● Attractiveness for everyone
  – Simple networking concepts
  – Low threshold
  – Availability in various platform and languages

● Technical requirements
  – Bidirectional, low latency, high bandwidth communications
  – Quick service connection/disconnection notifications
Real Applications

- Steerable Camera Projector pair
- 3D Tracker
- Interaction groups detector
- User level services
- Mapping project
- ...

29/06/2006 Rémi EMONET - INRIA Rhône-alpes