Towards Semantic Modelling in Adaptive Ubiquitous Environments

Tom Gross
Faculty of Media
Bauhaus-University Weimar
Bauhausstr. 11, 99423 Weimar, Germany
tom.gross(at)medien.uni-weimar.de

ABSTRACT
In this paper we glance at our current work on the concept and implementation of a service-oriented sensor-based platform to support adaptive ubiquitous environments and motivate the interest in this workshop on semantic models for adaptive interactive systems.

Author Keywords
Computer-Supported Cooperative Work; Context Modelling; Semantic Modelling.

ACM Classification Keywords
H.5.2 [Information Interfaces and Presentation]: User Interfaces — Graphical User Interfaces, User-Centred Design; H.5.3 [Information Interfaces and Presentation]: Group and Organisation Interfaces — Computer-Supported Cooperative Work.

INTRODUCTION
In our Cooperative Media Lab we develop adaptive cooperative ubiquitous environments supporting easy communication and cooperation within collocated groups, and among distributed sub-groups in mixed reality settings. We follow a layered approach in several steps: we first developed the based platform, then extended the platform, and now we integrate more and more inference engines.

SENS-ATION
Sens-ation is a platform that provides developers of sensor-based infrastructures with a toolkit and building blocks for the infrastructures they want to create and maintain [5]. It follows the tradition of examples such as Khronika [9], Elvin [4], and ENI [8]. And it offers sensors for capturing data from the real word and from the electronic world; persistence and querying for storing the data and inferring on the data, and clients for retrieving the data. Its main advantage is the abstraction from basic hardware, software, and network technology.

SensBution [7] is an extension of Sens-ation [5] for a peer-to-peer architecture based on services. With SensBution—and its multifarious adaptors for integrating sensors, and gateways for integrating clients and peers—developers can create and maintain distributed infrastructures rather easily. Here we just want to give an overall impression (Figure 1 shows the components of a SensBution peer). The basic components of SensBution are peers consisting of several subsystems and components.

- The *Adapter* subsystem connects sensors to the respective SensBution peer and submits sensor event data via Web Services, XML-RPC, common gateway interface, hypertext transfer protocol, or socket connection.
- The *SensorPort* component receives incoming sensor events from the adapters and forwards the events to the Management subsystem.
- The *Management* subsystem consists of the SensorHandler, the Registry, and the GatewayHandler component. The SensorHandler processes the sensor input received via the SensorPort. The Registry maintains data on the resources of the platform, such as active sensors, or locations. The GatewayHandler processes requests from the gateway components.
- The *Publisher* component offers a publish-subscribe mechanism that can be accessed via remote method invocation of the RMI component in the Gateway subsystem.
- The *Inferencing* subsystem provides an interface for inference engines that process and infer on sensor event data.
- The *Persistence* subsystem consists of the Database (long-term storage) and the Cache (short-term storage) component.
- The *RuleInferencing* subsystem handles and processes rule-based queries. Its components are: the QueryHandler, the QueryProcessing, and the Rulebase. The *QueryHandler* component handles incoming queries from the Gateway subsystem. If the query comes from another peer, the QueryHandler forwards the query to *QueryProcessing* and gets the results immediately. If the query comes from a client, the QueryHandler forwards the query to the Gateway P2P subsystem for distribution in the peer-to-peer network, and then forwards the query to QueryProcessing. The results are delivered to the requesting Gateway. The QueryHandler incorporates a query scheduler. The QueryProcessing component extracts all parts of received queries and forwards them to the Rulebase for evaluation. The result received by the Rulebase is handed back to the

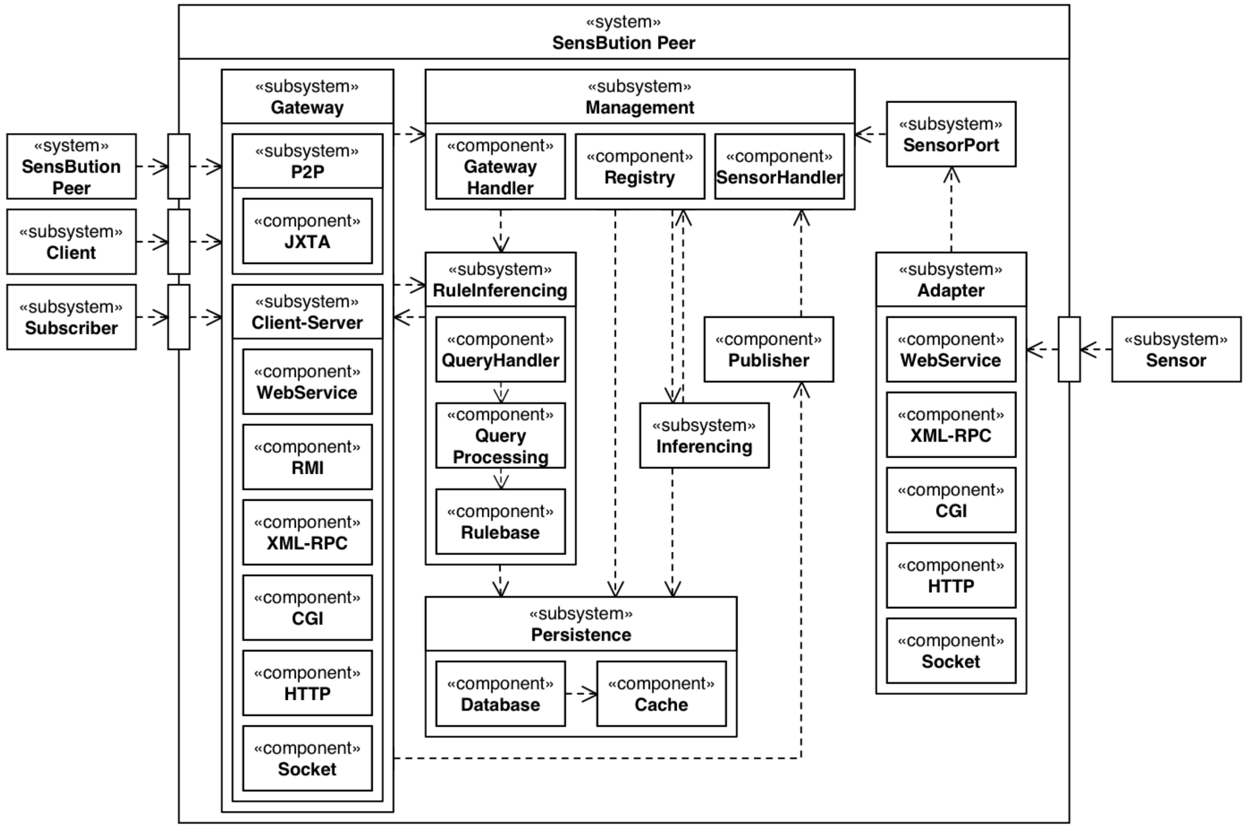

Figure 1. Components of a SensBution peer. Source: [7].

QueryHandler. The Rulebase maintains all rules and supplies the derivation rules delivering the query results.

- The *Gateway* subsystem handles requests and responses over different kinds of protocols and interfaces (i.e., a peer-to-peer gateway, and client-server gateways for Web Services, remote method invocation, XML-RPC, common gateway interface, hypertext transfer protocol, and sockets). The Gateway receives requests and forwards each of them by the GatewayHandler to the responsible subsystems. The *Gateway P2P* subsystem distributes queries to and from other SensBution peers. Queries get distributed according to the one-to-many or broadcast pattern—therefore, SensBution does not need a no central routing instance to direct queries to peers. Each SensBution peer receives the queries of all other SensBution peers in the same peer group. The Gateway P2P subsystem sends answers to the querying peer by a one-to-one pattern since the addressee is known.

Sens-ation and SensBution were developed with Java 2 Standard Edition 5.0 Platform [10]. The JXTA [11] protocols are used for the peer-to-peer implementation, based on the abstract programmers interface of the JXTA Abstraction Layer (JAL). The inference engines are based on Prolog [2] algorithms; and the Mandarax Java framework is used for derivation rules and an implementation of a rule engine [3].

DEALING WITH THE SEMANTICS OF EVENT DATA

Sens-ation and SensBution offer generic programming interfaces for implementing adaptation into inference engines that process the event data captured by the sensors and stored in the platform.

The event data that is captured by the sensors is represented as attribute-value pairs. Each event has mandatory attributes that are required—that is, each event that is captured by sensors needs values for these attributes:

- SensorID
- SensorType
- SensorValue
- OccurrenceDate
- OccurenceTime
- Location

Each event has optional attributes—that is, here attributes have standardised labels, but do not necessarily need values in each single event data:

- UserList
- RelativeTimestamp
- Urgency
- Sampling

- Frequency
- Granularity
- Ingredients
- Relationship

Finally, events can have any number of custom attributes. Custom attributes consist of a string representing the key, and a string or number representing the value.

In Sens-ation and SensBution all components that do operations on event data are called inference engines; they range from rather simple operations to machine learning algorithms. Some basic inference engines allow the processing of numerical data (e.g., min, max, average) and of text data (e.g., pattern matching). Some more complex inference engines are based on machine learning techniques.

For instance, the CoDaMine engine does communication data mining: it performs supervised learning on the text chat contents of instant messaging users in order to learn about their current situation and adapt the environment accordingly [6].

Currently a recommender engine is integrated into Sens-ation with the inference engine programming interface. We use the Apache Mahout Taste that provides a flexible and fast recommender engine [1]. It offers a data model and storage, and it provides user similarity and item similarity. We use it for storing user settings and preferences of our sensor-based environments and for providing suggestions for adapting the environment to the group of present users.

CONCLUSIONS

The topics suggested in the workshop call for position papers are very interesting. In the workshop I would be particularly interested in discussing semantic modelling approaches for adaptive cooperative ubiquitous environments that combine the strengths and provide synergies between modelling approaches from computer-supported cooperative work and ubiquitous computing with concepts from semantic modelling.

BIOGRAPHICAL INFORMATION

Tom Gross is associate professor for Computer-Supported Cooperative Work and head of the Cooperative Media Lab at the Faculty of Media of the Bauhaus-University Weimar, Germany. His research interests include Computer-Supported Cooperative Work, Human-Computer Interaction, and Ubiquitous Computing. Since beginning of 2008 he is Prorektor (vice-president) of the Bauhaus-University Weimar. He holds a diploma and a doctorate degree in Applied Computer Science from the Johannes Kepler University Linz, Austria.

ACKNOWLEDGEMENTS

Thanks to the members of the Cooperative Media Lab—especially Mirko Fetter, and Thilo Paul-Stueve. Part of the work has been funded by the Federal Ministry of Transport, Building, and Urban Affairs and by the Project Management Juelich (TransKoop FKZ 03WWTH018).

REFERENCES

[1] Apache Software Foundation. *Apache Mahout - Taste Documentation*. http://lucene.apache.org/mahout/taste.html, 2009. (Accessed 1/12/2009).

[2] Colmerauer, A. and Roussel, P. The Birth of Prolog. In *Second ACM SIGPLAN Conference on History of Programming Languages - HOPL'92* (Apr. 20–23, Cambridge, MA). ACM Press, New York, NY, 1992. pp. 37–52.

[3] Dietrich, J. *The Mandarax Project*. Open Source Technology Group, http://mandarax.sourceforge.net/, 2004. (Accessed 7/3/2007).

[4] Fitzpatrick, G., Kaplan, S., Mansfield, T., Arnold, D. and Segall, B. Supporting Public Availability and Accessibility with Elvin: Experiences and Reflections. *Computer Supported Cooperative Work: The Journal of Collaborative Computing 11*, 3-4 (2002). pp. 447-474.

[5] Gross, T., Egla, T. and Marquardt, N. Sens-ation: A Service-Oriented Platform for Developing Sensor-Based Infrastructures. *International Journal of Internet Protocol Technology (IJIPT) 1*, 3 (2006). pp. 159-167.

[6] Gross, T. and Fetter, M. CoDaMine: Communication Data Mining for Feedback and Control in Ubiquitous Environments. In *Proceedings of the Sixteenth Euromicro Conference on Parallel, Distributed, and Network-Based Processing - PDP 2008* (Feb. 13-15, Toulouse, France). IEEE Computer Society Press, Los Alamitos, 2008. pp. 539-546.

[7] Gross, T., Paul-Stueve, T. and Palakarska, T. SensBution: A Rule-Based Peer-to-Peer Approach for Sensor-Based Infrastructures. In *Proceedings of the 33rd EUROMICRO Conference on Software Engineering and Advanced Applications - SEAA 2007* (Aug. 27-31, Luebeck, Germany). IEEE Computer Society Press, Los Alamitos, 2007. pp. 333-340.

[8] Gross, T. and Prinz, W. Modelling Shared Contexts in Cooperative Environments: Concept, Implementation, and Evaluation. *Computer Supported Cooperative Work: The Journal of Collaborative Computing 13*, 3-4 (Aug. 2004). pp. 283-303.

[9] Loevstrand, L. Being Selectively Aware with the Khronika System. In *Proceedings of the Second European Conference on Computer-Supported Cooperative Work - ECSCW'91* (Sept. 24-27, Amsterdam, NL). Kluwer Academic Publishers, Dortrecht, NL, 1991. pp. 265-278.

[10] Sun Microsystems, I. *J2SE 5.0*. http://java.sun.com/j2se/1.5.0/, 2007. (Accessed 15/3/2007).

[11] Sun Microsystems Inc. *jxta: JXTA Commnity Projects*. https://jxta.dev.java.net/, 2010. (Accessed 8/1/2010).