Poster Abstract: SpyGlass: Taking a closer look at Sensor Networks

Carsten Buschmann
Institute of Operating Systems and Computer Networks
Technical University at Brunswick
D-38106 Braunschweig, Germany
+49 531 391-3289
buschmann@ibr.cs.tu-bs.de

Dennis Pfisterer
Institute of Operating Systems and Computer Networks
Technical University at Brunswick
D-38106 Braunschweig, Germany
+49 531 391-3295
pfisterer@ibr.cs.tu-bs.de

Stefan Fischer
Institute of Operating Systems and Computer Networks
Technical University at Brunswick
D-38106 Braunschweig, Germany
+49 531 391-3294
fischer@ibr.cs.tu-bs.de

Sándor P. Fekete
Department of Mathematical Optimization
Technical University at Brunswick
D-38106 Braunschweig, Germany
+49 531 391-7551
s.fekete@tu-bs.de

Alexander Kröller
Department of Mathematical Optimization
Technical University at Brunswick
D-38106 Braunschweig, Germany
+49 531 391-7410
a.kroeller@tu-bs.de

ABSTRACT
In this paper we present a modular and extensible visualization framework for wireless sensor networks. These networks have typically no means of visualizing their state, measurements or computational results. Visualization is therefore a key issue to develop and operate these networks. Data emitted by individual sensor nodes is collected by the visualization software and passed to a flexible multi-layer plug-in mechanism that renders the information on a canvas. Developers can easily adapt existing or develop new custom tailored plug-ins for their specific application.

Categories and Subject Descriptors
C.2.3 [**Network Operations**]: *Network monitoring.*

General Terms
Measurement, Performance, Experimentation.

Keywords
Sensor networks, smart dust, visualization, debugging, embedded computing.

1. INTRODUCTION
Recently, the study of wireless sensor networks [4] has become a rapidly developing research area that offers fascinating perspectives for combining technical progress with new applications of distributed computing. Nevertheless, the inherent lack of user interfaces leads to a tedious and error-prone developing and debugging cycle for which embedded systems are ill-reputed. In addition, unpredictable hardware and communication behaviors aggravate application development. While there is tool-chain support for embedded systems, development software for sensor networks is still in the early beginnings.

With the SpyGlass sensor network visualizer we aim at easing the life for sensor network debugging, evaluation and deeper understanding of the software by visualizing the sensor network, its topology, the state and the sensed data. A few tools exist which cover some aspects of sensor network data display and visualization. The feature-richest among them are the *Surge Network Viewer,* which is part of the *TinyOS* [5] project, and Crossbow's *TASK Environmental Monitoring Software [6]*. The *Surge Network Viewer* visualizes the topology of the network and features logging as well as viewing of network statistics and logged data. *TASK Environmental Monitoring* Software tries to flatten the learning curve for non-computer-scientists by simplifying the data collection, retrieval and visualization process using a database-like interface. Our approach differs from these tools in its independency from the sensor network hardware and its easy extensibility by implementing plug-in components.

2. SPYGLASS ARCHITECTURE
The visualization framework consists of three major functional entities: The sensor network, the gateway nodes located in the sensor network and the visualization software. Each individual sensor may collect data or derive it from calculations and forward it to a gateway node. Gateway nodes form the transducer from the wireless technology used in the sensor network to some TCP/IP enabled transit network (e.g. LAN, WLAN, GPRS, etc.). They also store a number of received data packets in a circular buffer to have some memory of past events. This enables them to bridge the time gap of transit network failures or to provide data to

SenSys'04, November 3–5, 2004, Baltimore, Maryland, USA.
ACM 1-58113-879-2/04/0011.

visualization stations which connect at a later point in time. Once a visualization station is connected to the gateway node it will receive all stored and future packets.

Data packets carrying arbitrary information (e.g. sensor readings, calculated data, internal sensor state, etc.) consist of a data type indicator, length information and the data. Using this simple format, developers can come up with new data types which will be immediately supported by the sensor network and the gateway software. Up to this point the data has only been forwarded, all data processing and display tasks are performed by the visualization software. Using this architecture and data format makes it possible to replace each of the three components individually, since the communication between them follows a well-defined packet format. Currently we provide two sensor network configurations: a real life sensor network using the embedded sensor board ESB 430/2 [1] and data originating from the ns-2 [2] network simulator.

Once the data has been delivered to one or more visualization stations the real processing takes place. After duplicates have been sorted out, plug-in components inspect the incoming data packets, extract data and infer the sensor network state. Five plug-in types for specialized tasks exist:

- *Node Painter plug-ins* draw the actual nodes and additional information onto the canvas. The depiction may be dependent on the node type (e.g. gateway node, cluster head, etc.).
- *Node Relation Painter plug-ins* display arbitrary relations between sensor nodes onto the canvas, e.g. by using lines to connect related nodes. Such might be communication links, group membership or routing paths.
- *Background Painter plug-ins* draw the background of the visualization canvas. They can also be used to illustrate spatial phenomena which can be inferred from the received sensor data and positions. Examples are temperature maps [3] or the display of coordinate systems.
- *Node Positioner plug-ins* are used by *Node Painter and Node Relation Painter plug-ins* to determine where to paint the nodes and relations on the canvas. Placement decisions can be either based on location estimates/measurements received from the sensor network or on strategies based on graph theoretical calculations optimizing screen representation.
- *Global information plug-ins* display information about partitions or the whole network in a textual way. This information is displayed in a sidebar and can be structured in a tree. Examples are the overall number of nodes, average neighborhood degree, etc.

For each plug-in type there is a container which holds one or more plug-ins of the corresponding type. The three canvas drawing plug-in types (*Node Painter, Relation and Background plug-ins)* have their own layer on the canvas on which their plug-ins exclusively draw. The user can change the order and visibility of the plug-ins within each plug-in container to adapt the presentation. The presentation is not limited to just on screen display; it can also be rendered to a user defined canvas (e.g. Postscript, MJPEG, etc.). Another feature of the visualization software is recording and playback of visualization sessions.

Adding an additional sensor reading to the visualization is straightforward. To implement for example the visualization of motion detection sensor readings, a developer simply needs to define a new data type, construct it as a binary array in the sensor and use the common forwarding service to a gateway node. To visualize the data, a simple *Node Painter plug-in* must be implemented which simply registers itself as a handler for the new data type, parses it and attaches this information to the corresponding node.

3. STATUS AND FUTURE WORK

We are currently finishing and testing the implementation and plan to publish it under an open source license soon. The implementation includes plug-ins for topology information, sensor node positions, temperature maps and sensor readings as well as general information (e.g. overall node count, average neighborhood degree, etc.). In future work we will to implement support for the Berkeley Motes hardware platform, 3D visualization and additional plug-ins.

4. ACKNOWLEDGMENTS

This work is part of the SWARMS and SwarmNet projects funded by the German Research Foundation (DFG). For further information, see http://www.swarms.de and http://www.swarmnet.de.

5. REFERENCES

[1] Website of the Embedded Sensor Board ESB 430/2: http://www.scatterweb.com.

[2] The Network Simulator - ns-2: http://www.isi.edu/nsnam/ns.

[3] C. Buschmann. D. Pfisterer and S. Fischer: Experimenting with Computer Swarms: a Mobile Platform based on Blimps, Poster, The Second International Conference on Mobile Systems, Applications, and Services, June 2004.

[4] I.F. Akyildiz, S. Su, Y. Sankarasubramanian and E. Cayirci: Wireless Sensor Networks; A Survey, Computer Networks, Vol. 38, No. 4, March 2002, pp. 393 – 422.

[5] TinyOS: http://www.tinyos.net.

[6] TASK Environmental Monitoring Software, Crossbow Technology Inc.: http://www.xbow.com/Products/productsdetails.aspx?sid=88.