# SASSI: The Sliverware Architecture for Sensor System Integration

Seth Holloway
Alexander Griffith
Angela Dalton
Drew Stovall
Christine Julien

TR-UTEDGE-2007-011

# SASSI: The Sliverware Architecture for Sensor System Integration

Seth Holloway, Alexander Griffith, Angela Dalton, Drew Stovall, Christine Julien
Department of Electrical and Computer Engineering
The University of Texas at Austin

{sethh,griffy2008,adalton,dstovall,c.julien}@mail.utexas.edu

## Categories and Subject Descriptors

D.2 [**Software Engineering**]: Software Architectures; D.2.11 [**Software Engineering**]: Software Architectures— *data abstraction, domain-specific architectures*

## General Terms

Algorithms, Design, Standardization, Theory

## Keywords

Sliverware, Sensor Middleware, Component-based Development

## 1 Introduction

Recently, embedded sensor usage has increased thanks to the proliferation of hardware and software addressing resource constraints. While the increased usage is a good start, it is important to adopt good software engineering principles early for many reasons: simplified programming and deployment, improved reuse, and added efficiency. This poster presents SASSI, the Sliverware Architecture for Sensor System Integration, which provides a unique embedded sensor application architecture [1].

SASSI builds on our previous research into component-based middleware, Sliverware [2]. Sliverware abstracts application support into three layers: the network communication layer, the group membership layer, and the services layer. Using such a layered approach, software developers are automatically aided by good design practices for structuring their code. We define a *sliver* as a combination of one Sliverware component from each of the three layers. Slivers define useful application stacks that can easily be reused or extended by programmers of varying expertise.

While there has been a great deal of work on embedded sensors and applications for them, there are few architectures that provide the ability to reuse existing components in a highly flexible manner and unify development. One similar effort is SNACK [1], which provides nesC component libraries for building applications using its own configuration language. The primary goal of SNACK is the construction

---

of efficient applications through integration of components from its library. Another more recent project, OASiS [3], is a service-oriented architecture that, through the composition of service modules, allows application development at a high level. Sliverware's use-only-what-is-needed approach helps keep overhead low and execution environments small which makes Sliverware ideal for embedded devices that cannot support bloated middleware. Additionally, the simplicity of creating high level applications is particularly beneficial because of the wide range of domains in which sensor-based applications are being adopted. SASSI gives domain experts the ability to formulate their own set of applications for a sensor deployment.

## 2 The SASSI Approach

As a concrete example illustrating the benefits of building embedded sensor applications using SASSI, we are prototyping a SASSI Aware Home. Development of the home is currently underway; the prototype physically occupies a 10'x16' mock home space in our lab. Our goal is to minimize overall power consumption while maintaining a comfortable environment for occupants. We achieve this by sensing occupants' location and controlling the temperature in those areas using a range of hardware including laptops and SunSpot sensors. The overall functionality is built upon three individual sensor application stacks, or slivers, integrated using SASSI. Figure 1 illustrates our aware home software structure and will be used to explain salient points about SASSI.

Sliverware's cross-cutting vertical components, slivers, define collaboration components. A single sliver provides a single high-level collaboration function (e.g., presence information about a homes occupants).

Slivers are made up of three layers: collaborative services, group membership, and network communication. The collaborative service layer (e.g., location service) hides complex, low-level coordination function by encapsulating changes within abstract events. The group membership layer (e.g., house presence group) coordinates related groups of devices based on application-specified policies. The network communication layer (e.g., RFID) provides communication capabilities that adhere to Sliverware-provided interface.

One or more slivers are used to create an application. A single application is often more complex than a single sliver; an application developer may require multiple collaborative activities (e.g., presence information and temperature events

to open and close air conditioning vents). The ability to combine slivers in this way makes applications highly modular.

Due to the inherent modularity, SASSI allows multiple applications to be integrated. In embedded sensor applications, this allows a unified interface to a wide set of functionality (e.g., applications to control a home).

Developers tailor slivers and SASSI applications in three ways: creating new applications using existing slivers; creating new slivers by combining existing sliver components; introducing new low-level capabilities by defining new sliver components. This allows developers with different skill sets to contribute modules that can be efficiently incorporated into applications. At the lowest layer are Sliverware component developers who are above-average programmers that possess layer-specific knowledge (communication protocols/coordination/services). The next level of Sliverware programming uses these components to create new slivers. In this stage we imagine programmers who, with no explicit knowledge of the low-level details, combine components to create useful applications. For example, a sensor-based application may use any one of the many networking protocols; however, it is not necessary for the sliver creator to know how to implement the protocols (e.g., multicast or flooding). Instead, he must understand the possible domains and adapt the components accordingly (i.e., an embedded sensor application should not use standard communication flooding due to the high energy overhead). The highest and most straightforward level of Sliverware programming is intended to be accessible by anyone with some knowledge of computers and their intended domain. The sliverware application developer simply takes an existing sliver and calls the service layer to accomplish the necessary tasks.

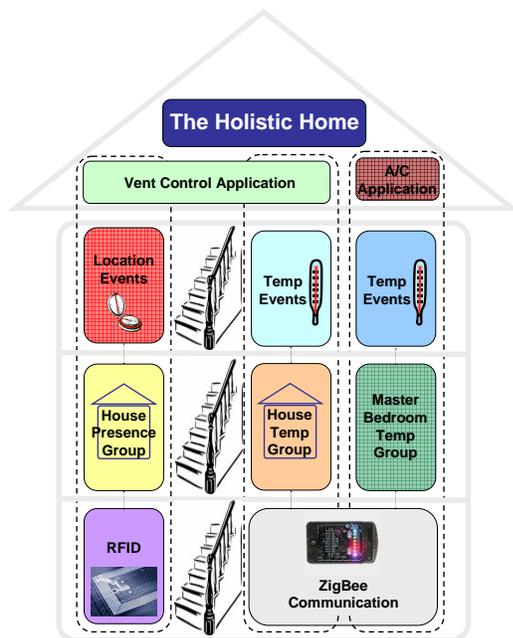Sliver components are easily maintained and upgraded,



**Figure 1. SASSI Aware Home Diagram.**

which is essential because sensing and communication technologies are rapidly changing. New technologies often differ significantly from previous generations. Sliverware components are separated by crisply defined interfaces. At any layer, a new implementation of a sliver component can be swapped in for an outdated component with no impact on the remainder of the sliver or on application functionality. If an application was using 802.11, it could instead use ZigBee by swapping in a ZigBee component in place of the 802.11 component.

Redundancy in slivers and applications is handled within the Sliverware infrastructure. Embedded sensor applications typically operate independently, often resulting in significant amounts of waste due to redundant sensing and communication across applications. The Sliverware infrastructure merges slivers bottom-up and combines functionality that is redundant across seemingly independent applications (e.g., reuse already deployed communication protocols or a previously defined group).

## 3 Conclusion

The Sliverware approach has a number of key benefits for embedded sensor systems. Specifically, SASSI:

- Provides component reusability on-the-fly,

- Permits easy upgradability and maintainability,

- Enables application integration,

- Allows rapid application development and tailoring at varying levels of expertise,

- Promotes abstraction by hiding low-level complexity, and

- Drives separation of concerns by separating communication and sensing from application behavior.

Our example application, the SASSI Aware Home, minimizes power consumption while maximizing occupants' comfort. Our prototype implementation contains several types of hardware including laptops and SunSpot sensors to demonstrate feasibility. By using Sliverware in this setting, we have created a solid foundation for future work and provided an implementation with several different sensors and integrated applications.

## 4 References

[1] B. Greenstein, E. Kohler, and D. Estrin. A sensor network application construction kit (snack). In *Proc. of the 2nd Int'l. Conf. on Embedded networked sensor systems*, 2004.

[2] S. Holloway and C. Julien. Developing collaborative applications using sliverware. In *Proc. of the 14th Int'l. Conf. on Cooperative Information Systems*, 2006.

[3] M. Kushwaha, I. Amundson, X. Koutsoukos, S. Neema, and J. Sztipanovits. Oasis: A programming framework for service-oriented sensor networks. In *IEEE/Create-Net COMSWARE*, January 2007.