A Reflective Middleware Architecture for Distributed Sensor Applications

Qi Han
Department of Mathematical and Computer Sciences
Colorado School of Mines, Golden, CO
qhan@mines.edu

ABSTRACT

To enable a wider adoption of sensor network technologies, we must address a variety of constraints inherent in sensor network operation and provide a significantly rich level of abstraction to application users supported by efficient and robust optimization and evaluation techniques. In this paper, we develop a reflective middleware system in order to support distributed sensor applications with various performance requirements. We explore the possibility, mechanisms and benefits of applying the reflection concept to distributed sensor environments. We will also implement a prototype system and validate it using a transient subsurface contaminant tracking application.

Categories and Subject Descriptors

C.2.1 [**Network Architecture and Design**]: Wireless communication; C.2.4 [**Distributed Systems**]: Distributed Applications

General Terms

Design, Management

Keywords

Reflective middleware, Wireless sensor networks

1. INTRODUCTION

In the coming years, large-scale wireless sensor networks will become a reality. In addition to monitoring applications that keep track of current sensor values, sensor network technologies can also be used for archival applications that store historical data (e.g., in order to detect patterns over time and build time-varying models), and forecasting applications that are interested in predicting future sensor values. A sensor network is often deployed for a particular application, which can potentially lead to a large number of application-specific protocols and systems. We, however, observe that sensor data collection is a common need for monitoring, archival or forecasting applications. Even though significant progress has been made towards in-network aggregation and filtering, many applications would still like to receive as much detailed data as possible from the sensor network. We also observe that many applications pose various performance goals in QoS (Quality of Service), QoD (Quality of Data) and Cost (i.e., energy consumption). QoS requirements are often specified in terms of response timeliness, resilience to failures, and QoD requirements refer to data accuracy or freshness. While some efforts have been invested in achieving a subset of the performance goals, dealing with diverse and changing performance goals in a systematic manner remains largely unsolved.

Wireless sensor networks have typically been built with a high degree of dependency between applications and the underlying communication protocols. Such dependency is justified as necessary to achieve energy efficiency. However, it generates rigid systems with sensor networks specifically designed to suit a particular application. While providing a platform that accommodates all types of sensor applications is very difficult, we intend to build a middleware architecture that can support a representative class of sensor We will investigate fundamental techniques that support sensor applications in a reflective middleware architecture, implement a prototype and validate it using a plume tracking application. By being reflective, the system is able to adapt its behavior based on its own observation of system status, so that the system can deal with changes in a non-intrusive manner as the system expands and evolves. While middleware typically hides the underlying details from applications and simplifies application development significantly, we argue that this transparency alone cannot deal with the high dynamism intrinsic in distributed sensor environments. Applications have valuable information that could enable the middleware to execute more efficiently under different application workloads and system statuses. Therefore, reflection offers significant advantages for building a sensor network middleware.

2. A REFLECTIVE SENSING ARCHITECTURE

Middleware, by its definition, hides the heterogeneity of the underlying layers, thus allowing application developers to focus on the important aspects of the application's logic. However, distributed sensor applications demand a high degree of flexibility and adaptability in order to deal with dynamic changes in application requirements and sensor envi-

ARM'06 November 27-December 1, 2006, Melbourne, Australia

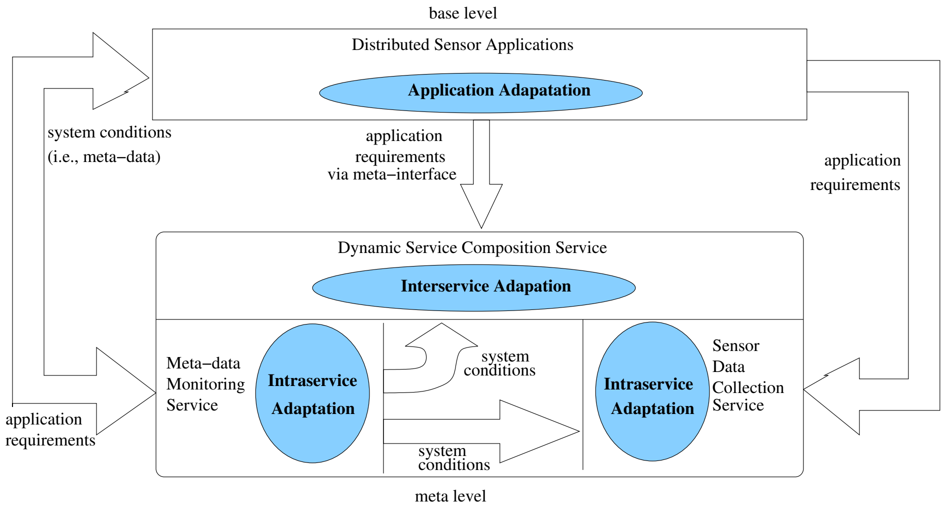

Figure 1: A reflective sensing architecture

ronments; hence, they can benefit greatly from knowing the status inside the underlying layers, and in the computational and physical environment. Therefore, we introduce the notion of computational reflection to the sensing architecture, bringing network and system monitoring support to the level of sensor applications. Computational reflection [2, 3] is a technique that allows a system to observe and maintain information about itself (meta-data) and use this information to change its behavior (adapt). In other words, the system maintains a causally-connected self representation. This is achieved by processing at two well-defined levels: functional level (also known as base or application level) and management (or meta) level. An important part of the reflection is the reification process - the capture and observation of the base level states.

Figure 1 is a reflective middleware architecture depicting the interaction and adaptation needed for the reification and reflection process. The system monitoring service (a meta-level service) provides fundamental techniques for keeping track of the network and application statuses and well-defined interfaces for the reflection process at the application layer. The data collection service (a meta-level service) is a library of data collection algorithms and network protocols that are adaptive to the dynamic meta-data provided by the monitoring service. The dynamic service composition service (a meta-level service) allows for composing meta-level services (i.e., data collection service and system monitoring service) for better adaptation to dynamic changes in system status.

Our reflective sensing architecture provides a structured methodology to support dynamic adaptation needed by sensor applications. In particular, adaptation is performed at both base and meta level.

- Base/application level adaptation refers to adaptation of the application by reacting to changes in system conditions [1]. The application may register its information request, such as attributes and adaptation parameters, to the meta level. Such awareness of system status allows the application to adapt proactively or to choose its application logic.
- Meta level adaptation consists of *inter-service* and *intra-service* adaptation. *inter-service adaptation* refers to adaptation of the middleware system by reconfiguring service composition. *intra-service adaptation* refers to adaptation of the middleware system by allowing the service to react to changes in meta-data. It is performed via an integration of (a) system-aware sensor data collection, where changes in system status can proactively trigger adaptations in data collection policies; and (b) application-aware system and network monitoring, where application requirements trigger optimization of monitoring policies and parameters.

3. EVALUATION

In addition to large scale simulation using the network simulator NS-2, plans are underway to implement a prototype of the proposed sensing architecture. We plan to validate the system through an environmental application - transient plume tracking. The reification and reflection capability of our system will significantly benefit the whole tracking process: adaptive network monitoring provides a fundamental support for sensor data collection; computation results based on collected sensor data provide a basis for calibration and validation of plume models. At the same time, more accurate identification and prediction of phenomena behavior can be used to drive more targeted sensor operations. The techniques developed in the paper will enable easy development of sensor applications using a common platform, which is essential to the overarching goal of true ubiquitous computing.

4. REFERENCES

[1] Q. Han, S. Gutierrez-Nolasco, and N. Venkatasubramanian. Reflective middleware for integrating network monitoring with adaptive object messaging. *IEEE Network*, 18(1), January 2004.

[2] P. Maes. Concepts and experiments in computational reflection. In *Proceedings of OOPSLA*, 1987.

[3] B. Smith. *Reflection and Semantics in a Procedural Language*. PhD thesis, Massachusetts Institute of Technology, January 1982.