The Design of an Open Data Service Architecture for Cyber-Physical Systems

Woochul Kang, Sang H. Son, {wk5f,son}@cs.virginia.edu
Department of Computer Science, University of Virginia

Abstract

In this paper, we discuss the data management issues in cyber-physical systems, and propose an open data service architecture to address the problem.

1 Introduction

In cyber-physical systems (CPS), data from wide-area sensors and data processing capacity of computing entities in Internet are combined, and the results are fed again into the physical world for actuation. We observe that CPS are inherently data-centric; the availability of fresh data at the right computation site at the right time is the very basic functionality that CPS require to interact with the physical world in real-time. However, we also observe that current computing- and communication-oriented paradigm poses a huge obstacle to achieve the vision of CPS.

The first problem is that there is no programming abstraction or protocol that enables efficient inter-operation and coordinated sharing of distributed and heterogeneous data. Since most data management schemes use ad-hoc data format and service interfaces, the inter-operation between them is very hard, if not impossible. For instance, most wireless sensor network (WSN) systems focus on wireless domain, and ignore how to interact seamlessly with external world. As a consequence, there is no open standard method to interact with WSN. Several wide-area sensor network architectures such as IrisNet [2] have been proposed, but inter-operations between heterogeneous data services have not been considered.

The second problem of the previous approaches is that they have not enough support to guarantee *Quality of Data (QoD)* (e.g., data freshness and correctness) and *Quality of Service (QoS)* (e.g., deadline miss ratio for transactions). QoD and QoS are supported in part, but not in an integrated manner by the architecture. In particular, even though several open standards are available for data access and management in Internet-scale computing platforms, they are scientific/business application-oriented and lack the consideration on QoS and QoD; an example is *data service architecture* for *Open Grid Service Architecture* (OGSA) [1].

CPS require the interaction between these heterogeneous and incompatible computing domains. The limitation of each computing domain demands an open data service architecture that is general enough to support the requirement of different computing domains and facilitates efficient inter-operation between them.

2 Open data service for CPS

2.1 Architecture

A data service is a functional unit which collects, stores, processes, and exports data. We propose an open data service architecture that consists of layers of programming abstractions as shown in Figure 1. The layered architecture enables the separation of concerns between layers and provides programming interfaces for different degrees of abstraction on the data service. For instance, a programmer who wants more control on the data service may use low level interfaces of the data service instead of using a high level interface, which is designed for more general purpose applications. Note that we choose to use web service as our communication protocol and all interfaces of the layers are accessible and defined by the web service framework. Since web protocol is supported by a wide spectrum of vertical and horizontal computing platforms, the choice of web service as a communication protocol will enable efficient inter-operation between them; in terms of WSN, gateways or back-end servers act as a proxy data server since sensor nodes are not powerful enough to run web service-based data services.

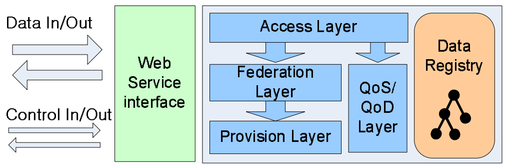

Figure 1: The layered architecture of the open data service.

The following explains the role of each layer.

1. *Access layer:* this layer provides common data access interfaces such as SQL. The declarative query language in this layer supports extended operations against streams since most data from sensors are continuous streams. The attributes and status of the service such as the target deadline miss ratio and the required data freshness can also be specified in this layer. Applications can control the behavior of a data service by interfacing this layer.
2. *Federation layer:* this layer enables the coordinated management of data in distributed sites. Since each data service is accessible with open interfaces defined by Web Service Definition Language (WSDL), the coordination and sharing data among data services can be achieved by composing distributed data services.
3. *Provision layer:* this layer provides the most primitive functions that are required to achieve the requirements

of the higher layers. The basic services provided in this layer include data transfer between distributed sites, data replication and coherence management, data filtering, storage space management and other lower level functions.

4. *QoS/QoD enforcement layer:* this layer enforces the QoS/QoD requirements specified in the access layer by interacting with the federation layer and the provision layer. The implementation of this layer may vary according to the capability of the federation and provision layers. For instance, for a local data service, QoS/QoD may be enforced by controlling local resources. However, for a data service that federates dispersed data sources in wide-area, the implementation of this layer may require dynamic caching and filtering of data to meet the performance goals. The QoS/QoD specification from the access layer can be guaranteed only if the QoS/QoD enforcement layer ensures that the federation and provision layers can provide enough resources to achieve them. For instance, if these resources are dispersed in wide-area, QoS/QoD enforcement layer should exploit the federation layer, which in turn executes resource negotiation process with distributed data services to secure resources to guarantee the desired QoS/QoD.

In our data service architecture, relational data model is assumed for its wide-spread use. Data streams are converted to relations and vice versa as in *Continuous Query Language*. Each data object is identifiable by globally unique identifier such as *end point references (EPR)* in the web service framework. In Figure 1, the data registry service is responsible for registering, publishing, updating, and finding data in the data service. Each data service is characterized by data objects it manages with certain QoS and QoD guarantees.

2.2 An Example Scenario

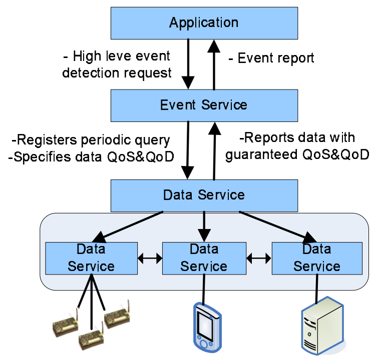

Figure 2: An event detection service in wide-area.

As a basic building block of CPS, the proposed open data service can be used to implement high level services. Figure 2 shows an example of a data service that facilitates event detection in wide-area. Instead of interacting directly with a number of distributed data sources, the event detection service interacts with a single data service that federates distributed data sources by coordinating distributed data services, which in turn manage their own data sources.

Note that even though underlying implementations of the data service can be different, they are accessed by the common open data service interface. Moreover, the open data service enables the data federation and sharing in wide-area with guaranteed QoS and QoD. The federation process is initiated by an application or a middleware by submitting a specification to the data service, which includes the data objects it wants to subscribe. The federation layer of the data service contacts data services which manage each data object to reach a service agreement on QoS and QoD of the service.

2.3 Implementation Issues

As an open data service architecture, each layer can be implemented differently as far as the implementation conforms to the constraints that each layer needs to satisfy. However, our data service architecture facilitates specific implementation decisions more naturally than others. For instance, since data service composition through federation layers enables hierarchical data-flow from data sources to sinks, hierarchical feedback control to guarantee QoS and QoD becomes a natural choice. Each common ancestor in the hierarchy may control the data transmission rate of children data services by adjusting filtering degree. It may need *multiple-input-multiple-output (MIMO)* control solutions to manage the QoS/QoD on multiple children data services simultaneously.

3 Future Research Direction

We believe that the proposed open data service architecture will be the basic building block of CPS, hence, enabling the seamless inter-operation between data services with guaranteed QoS/QoD support. Our future research agenda is as follows. First, we will work out a more detailed design to facilitate seamless inter-operation and architectural level support for QoS and QoD. Second, a simulator and a prototype will be implemented to show the validity and feasibility of the proposed architecture. Finally, we will invent techniques that will enable QoS and QoD guarantee in diverse computing platforms. For instance, guaranteeing the QoS and QoD in resource-constrained embedded systems and in a virtualized data service composed of distributed data sources may require totally different approaches.

References

[1] D. Berry, A. Luniewski, and M. Antonioletti. OGSA Data Architecture. OGF, September 2007.

[2] P. Gibbons, B. Karp, Y. Ke, S. Nath, and S. Seshan. IrisNet: An Architecture for a Worldwide Sensor Web. *IEEE Pervasive Computing*, 2(4), Oct-Dec 2003.