Dynamic data processing middleware for sensor networks

Teemu Leppänen and Jukka Riekki

Intelligent Systems Group and Infotech Oulu,
University of Oulu, P.O.Box 4500, 90014 University of Oulu,
Oulu, Finland

{Teemu.Leppanen, Jukka.Riekki}@ee.oulu.fi

Abstract. Heterogeneous and resource-constrained sensors, computational and communication latencies, variable geospatial deployment and diversity of applications set challenges for sensor network middleware. RESTful architecture principles have been widely applied in middleware design. The new Computational REST architecture offers additional set of principles. In Computational REST, computations are seen as resources and interactions are conducted as computational exchanges. In this PhD thesis work, these principles are studied and elaborated further in the context of sensor network middleware. Middleware and system component prototypes are developed, evaluated and utilized by field trials in real-world settings. As a result, new knowledge is generated of ubiquitous sensor network middleware design and dynamically distributing data processing computational load in resource-constrained sensor networks.

Keywords: sensor networks, middleware, computational REST, web of things

1 Introduction

Information and communication technology advancements accelerate the development and deployment of sensor networks, which are being utilized to collect, store and share information of different phenomena within environment and society today. Advantages of wireless sensor networks (WSN) over fixed sensor networks include dynamic deployment and changeable network topology [1]. Suitable platforms for heterogeneous sensor nodes include smart phones, PDAs, PCs, appliances in households, mobile data terminals and vehicle computing platforms. Also devices in public infrastructure, such as traffic signal poles, could be used as sensor node platforms. However, these devices have constrained resources in terms of computation, communication, memory, and battery life [1]. At the server side of the sensor networks, the sink nodes can be local or remote servers and even cloud computing platforms having virtually unlimited resources for data processing and storage.

Internet of Things interconnects everyday objects by using Internet protocols. The objects are sensors, devices and platforms, which then become Smart Things. Internet

of Things applications then utilize propriety communication protocols layered on top of Internet protocols such as TCP or UDP. The Web of Things has the advantage of using established Web standards with devices, which can have embedded Web servers running and integrate with any content from the Web [2]. The loosely-coupled nature of the Web and reusable common tools such as browsers, languages and interaction techniques can be applied to the real-world objects [2]. Along with the Web of Things, RESTful [3] architecture has been introduced and its principles are widely applied to the context of low-power WSN. RESTful architecture provides decentralized WSN with unified view of components, described for example in [2, 4]. Recent addition to the REST principles are the Computational REST (CREST) principles, suggested in [5]. The goals of CREST are the distribution of services away from the Web server and composition of higher-order services from established lower level services.

In our earlier research, we developed a sensor network middleware for traffic related sensor data collection and processing [6]. The system utilizes mobile phones with integrated sensors and external public services in Internet as data sources, using HTTP for data dissemination. This data is processed in real-time and results are visualized in an annotated map in a web browser. Moreover, we have developed a system for wireless monitoring of energy consumption of everyday electrical appliances in private households [7]. The wireless sensor nodes utilize 6LoWPAN protocol stack and HTTP for data dissemination to a centralized server, atop local IP-based panOULU WSN deployment. In the server, data can be visualized in a web browser and metrics of energy consumption are collected. Currently, we are advancing the panOULU WSN by planning new sensors, such as environmental sensors, and implementing REST interfaces for data dissemination in the system, internally between services and sensors, and externally with client applications.

This PhD thesis work continues our work on sensor data processing middleware. Based on our earlier research [6, 7], we have identified the following topics requiring further research: enabling seamless collection and processing of data in the sensor nodes, utilization of multiple heterogeneous data sources, data dissemination and persistent data storage, efficient distributed data processing at different levels of services and in multiple stages, data processing capabilities conditional on the data type, and controlling and monitoring the system behavior. A related issue is the lifespan of data, from milliseconds to years, within the applications where data type, quality and quantity are variable [6]. Studying these topics helps to develop middleware that can disseminate data in real-time, has low computational latencies, has energy efficient components, persistent data storage mechanism and has sufficient privacy and security features. Also, in this developed middleware, loosely-coupled architecture follows from the heterogeneous sensors, variable geospatial deployment and diversity of applications.

The rest of this paper is organized as follows. In Chapter 2, related work is given, Chapter 3 presents the PhD thesis work in more detail and Chapter 4 presents the discussion.

2 Related Work

In the RESTful architecture [3], clients connect to servers to access resources. A resource is any component or information that needs to be used and addressed. Resources are identified with URIs in well-known identification scheme and should be available through uniform interface, such as HTTP. Here, simple HTTP operations are used: GET, PUT, POST and DELETE. Interactions must be stateless, so that all the information needed to fulfill request must be part of the request. Services can be utilized through hypermedia links in resources or service discovery can be made using discovery protocols, described for example in [2, 4]. "Smart Gateways" [2] in the sensor network can also provide composition of low-level services as a higher-level service and in addition can be used for abstracting a non-RESTful service. This is useful for sensors which don't have enough capabilities to run a Web server. REST offers several advantages over traditional approaches [2, 4]. Interfaces in REST are simple and uniform. Implementations are light-weight and easily scalable as components can be completely independent. Lastly, all features already in HTTP are available for use.

In CREST [5], computations are seen as resources and interactions are conducted as computational exchanges. Computational exchanges are carried out in the form of continuations. A continuation is representation of the execution state of a computation, even including the required data, in a way it can be later resumed. CREST offers several extensions to the architectural principles of REST: 1) resource is an abstraction of a computation and named by URL, 2) representation of a resource is a program, closure, continuation or binding environment plus the metadata describing it, 3) each interaction contains all information necessary to complete the request, 4) few primitive operations are always available but resource specific operations are allowed for creating new representations, and 5) presence of intermediates is promoted for use. CREST URLs follow the structure: //server/url_0/…/url_{n-1}/data/ and are not intended to be human readable. The interpretation of CREST URLs is up to the binding environment and URLs can be freely modified by intermediates. A single service can be exposed through a number of URLs offering multiple perspectives to the computation which can be even recursive. In [8], the CREST architectures are taken one step further. As CREST component peers can be seen as clients and servers simultaneously, the concept of layers in middleware could be replaced with symmetric computational exchanges. Layers and services are a matter of convenience, perspective and scope [8]. This could question the concept of layered middleware altogether.

A number of sensor network middleware has been proposed in literature. The Devices Profile for Web Services suggested in [9] offers service-oriented architecture (SOA) middleware for WSN's. Middleware components run RESTful or optimized web service functionality to interconnect sensor nodes. Nodes connect to the network by plug-and-play, announce their presence and offered services in the network and publish/subscribe events to communicate. However, the SOA implementations can be too heavy for resource-constrained devices and do not truly expose the Smart Things functionality to the Web [2]. Comparisons made in [10] suggest that RESTful web services outperform SOAP based approaches in both request completion times and

power consumption. Another result is that when considering IPv6 addressing, request completion times in IPv6 are longer than in IPv4.

The DIGIHOME WSN middleware [4] for home monitoring systems supports REST interfaces for heterogeneous sensor nodes and computational entities, standard discovery and communication protocols, multiple resource representation formats and event-based reasoning. For the middleware, the authors introduce software connector encapsulation providing interactions between components, invocation, persistency and messaging. Component communication is event-based, events can be local or system-wide and event processors can trigger actions on nodes based situation changes. Default interaction protocol is HTTP because of its prevalence, also Web intermediates are used to modify data in transmission.

Recent example realizations of Web of Things were demonstrated in [2], where real-world devices with embedded HTTP servers, and Smart Gateways not supporting IP or HTTP, were implemented. Web mashups accessed the devices through RESTful interface, mapping the native requests, based on Bluetooth, for example, to URLs. The authors demonstrated mashup from the user interface in the computer to a physical device and mashup from a physical device to a device. Proxy servers were used for dynamically discovering mobile devices as sensor nodes and disseminating requests to them.

In the literature, other solutions have been described for these challenges. Sensor network can be "wired" for a task by configuration service or files. This wiring can also be routing or reasoning-based. For service discovery, protocols such as SLP, service discovery layers or agents can be used. For code migration, bytecode with virtual machines, migration agents or native code have been utilized. Possible data formats and representations include ASCII, JSON, HTML, XHTML, XML, binary encoded XML and even Java object serialization [2, 4, 10]. Built-in content negotiation feature in HTTP also simplifies the task of delivering data between components.

3 PhD Work

The objective of this PhD work is the development of scalable sensor network middleware and data processing architecture. The main features of the system will be 1) uniform interfaces for heterogeneous data sources and data dissemination between system components, 2) dynamic deployment of system components and 3) methods for distributing dynamically computational load in the sensor network. These features need to contribute towards energy efficiency. To achieve this, the granularity of system components needs to be studied: what are the component capabilities and what constraints sensors, data and services set in the system. Monitoring and dynamic controlling of the system will be studied, for example, by elaborating further the dynamic partitioning problem described in [11]. Tradeoffs between system features and requirements are inevitable, following from the variable device capabilities, system component configurations and diversity of applications. Another related open research question is the optimal number of sensors for an application.

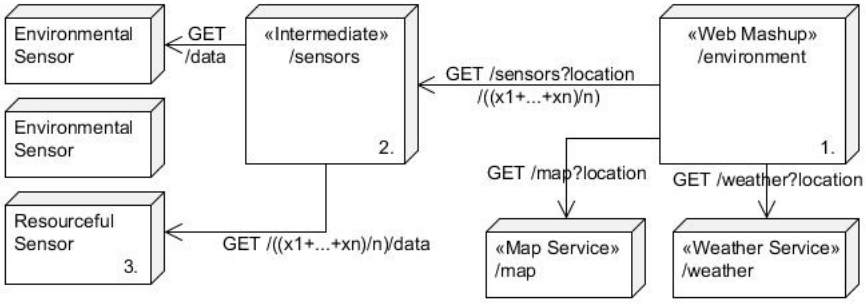

Fig 1. An example CREST architecture for an environmental monitoring system.

To the best of our knowledge, CREST has not yet been utilized in the context of sensor networks. In CREST, the origin server may not be responsible for all the computations as code migrates, thus the issue of computational latency is an important question. The system should have mechanisms such as concurrent processing and cost functions for minimizing the latency. One possible solution for this is the reuse of previously computed continuations [5]. To demonstrate how CREST can be applied in this work, a simple example architecture for an environmental service is sketched in Fig 1. Clients can access a Web mashup called /environment, integrating environmental sensor data with local weather data and visualizing it atop a map. An intermediate, called /sensors, is used to access the actual sensors. The mashup sends GET request with location parameters and a computation to the /sensors. This intermediate locates the corresponding sensor node and sends GET request to receive data from the node. Then the data is added to the URL, which is disseminated further to a resourceful sensor node for task completion (for calculating average in this case). When finished, computation results are returned to the mashup. In case this computation has already been partly computed, say weekly average till the previous day, a continuation can be used to reduce amount of required data and computational load. Also, the usage of intermediate is not required, but is used as an example to abstract sensors. In real-world, the intermediate could be for example a sink node or an edge router. The Web mashup /environment can itself be a continuation.

The middleware will be evaluated by having field trials in real life settings within the panOULU WSN and by using applications developed in research projects. In the evaluation, we are looking for answers to the following issues: 1) how much communication overhead is caused by the CREST principles and distributed processing, 2) are task or data query completion times reduced, 3) is the amount of data transferred in the network reduced and 4) can we reduce energy consumption in the resource-constrained parts of the network, such as sensor nodes. This work will be carried out in iterative manner based on literature reviews, implementation of middleware and component prototypes, tests and by collation of acquired experiences.

This PhD work generates new knowledge of ubiquitous sensor network middleware design and dynamic distribution of data processing load in resource-constrained sensor networks. Collected sensor data and processing results can be

made available immediately as RESTful web services and new applications can be deployed to use since the infrastructure is ready. Possible applications in the system include promoting sustainable energy consumption in private homes, real-time traffic data services integrated with people's movements and other environmental issues in urban areas can be addressed.

4 Discussion

Expected benefits of utilizing CREST principles with sensor networks include: reusable data and computations in the form of continuations, efficient usage of resources with load balancing and distributed processing, easy context-switching and simplified deployment of tasks, services or resources to the network simultaneously promoting scalability [5]. Even service discovery mechanisms may not be needed at all in the system and it is possible to describe binding environments in the URL [8]. Concerning the given characteristics and requirements from Chapter 1, the proposed middleware is loosely-coupled and allows dynamic variable geospatial deployment of system components. How the code migration is actually implemented is an open question, but a number of techniques have been proposed in the literature. Heterogeneous sensors can be used, provided that proxies or intermediates are developed, especially for resource-constrained parts in the network. Constrained Application Protocol (CoAP) [12] could be one solution to realize this kind of system, being a web protocol optimized for constrained environments and having block transfer capability for larger amounts of payload in the messages [13]. For privacy and security issues, HTTP already provides basic mechanisms and these can be studied further. Interesting question is the role of the middleware which can be diminished altogether by these features, as suggested in [8].

However, REST is not perfect solution for sensor networks. According to [2], REST architectures are not very suitable for real-time information and event-based or streaming applications because of HTTP limitations. Computational latencies in REST and communication overhead caused by HTTP are inevitable issues that need to be studied. One example of such systems is the collection and analysis of traffic data provided by mobile devices, where real-time communication needs to be encapsulated with Web intermediate for higher-level services. Possible solution for this would be real-time "streaming channel" to/from the entities in traffic, controlled through REST interface operations and perhaps utilizing CoAP for data dissemination. It is currently unknown how data will be stored in the system for long term analysis, but this issue is also application-specific. Additionally, CREST URLs can be of size of ten's of megabytes [5], which is not very suitable for low-power resource-constrained sensor nodes. This could be one serious limitation in the system, but it can be addressed by using intermediates or proxies. To what extent CREST works with low-power sensor nodes remains an open question.

Based on previous research, the sensor network infrastructure in urban areas can be utilized to collect information and create new services in public and residential areas. The middleware developed in this PhD work provides a platform for these services as well as contributes towards ubiquitous computing and Web of things.

Acknowledgements

This work is carried out in co-operation with Mediateam Oulu, University of Oulu. The authors would like to thank all the personnel in the RealUBI project. This work was funded by the Finnish Funding Agency for Technology and Innovation, the European Regional Development Fund, the City of Oulu and the Urban Interactions consortium.

References

1. Buratti, C., Conti, A., Dardari, D., Verdone, R.: An overview on wireless sensor networks technology and evolution. Sensors, 9, 9, 6869--6896 (2009)
2. Guinard, D., Trifa, V.: Towards the Web of Things: Web Mashups for Embedded Devices. In: WWW2009 Workshop on Mashups, Enterprise Mashups and Lightweight Composition on the Web (2009)
3. Fielding, R.: Architectural styles and the design of network-based software architectures. Dissertation, University of California, Irvine (2000)
4. Romero, D., Hermosillo, G., Taherkordi, A., Nzekwa, R., Rouvoy, R., Eliassen, F.: RESTful Integration of Heterogeneous Devices in Pervasive Environments. In: 36th EUROMICRO Conference on Software Engineering and Advanced Applications, pp. 1--14. (2010)
5. Erenkrantz, J., Gorlick, M., Suryanarayana, G., Taylor, R.: From representations to computations: The evolution of web architectures. In: 6th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering, pp. 255--264. ACM, New York (2007)
6. Leppänen, T., Perttunen, M., Riekki, J., Kaipio, P.: Sensor Network Architecture for Cooperative Traffic Applications. In: 6th International Conference on Wireless and Mobile Communications, pp. 400--403. (2010)
7. Ojala, T., Närhi, P., Leppänen, T., Ylioja, J., Sasin, S., Shelby, Z.: UBI-AMI: Real-time metering of energy consumption at homes using multi-hop IP-based wireless sensor networks. LNCS Advances in Grid and Pervasive Computing, 6646, pp. 274--284. Springer (2011)
8. Gorlick, M., Erenkrantz, J., Taylor, R.: The Infrastructure of a Computational Web. Technical Report, University of California, Irvine (2010)
9. Abangar, H., Barnaghi, P., Moessner, K., Nnaemego, A., Balaskandan, K., Tafazoli, R.: A Service Oriented Middleware Architecture for Wireless Sensor Networks. In: Future Network & Mobile Summit (2010)
10. Yazar, D., Dunkels, A.: Efficient application integration in IP-based sensor networks. In: First ACM Workshop on Embedded Sensing Systems for Energy-Efficiency in Buildings, pp. 43--48. ACM, New York (2009)
11. Chun, B., Maniatis, P.: Dynamically partitioning applications between weak devices and clouds. In: First ACM Workshop on Mobile Cloud Computing & Services: Social Networks and Beyond, pp. 1--7. ACM, New York (2010)
12. Constrained Application Protocol, http://datatracker.ietf.org/doc/draft-ietf-core-coap/
13. Blockwise transfers in CoAP, http://datatracker.ietf.org/doc/draft-ietf-core-block/