International Journal of Database Theory and Application
Vol.9, No.4 (2016), pp. 119-134
http://dx.doi.org/10.14257/ijdta.2016.9.4.11

A Web-Based Framework for Lightweight Context-Aware Mobile Applications

Jianchao Luo[1] and Hao Feng[2]

[1] *School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu, 611731, China*
[2] *School of Computer Science and Engineering, Guilin University of Electronic Technology, Guilin, 541004, China*
luojc@uestc.edu.cn, fengh@guet.edu.cn

Abstract

As more and more intelligent sensors and sensing applications are equipped with, smartphones are becoming smarter and play a greater role in people's lives. However, due to mobile platform diversity, the development and deployment of a context-aware application for different mobile devices are time-consuming and expensive, which in fact limits the large-scale application of context-aware technology in mobile heterogeneous environments. Unlike native applications, web applications are easy to develop, upgrade and deploy, and almost all smartphones today include a web browser for supporting running them. Another difference is that native applications can capture context information by accessing sensors available on the mobile device, but web applications running in the browser cannot. In this paper, we describe CaMWAF, a framework designed to support the rapid development of context-aware mobile web applications and simplify the exchange of context information in heterogeneous environments. With CaMWAF, developing a context-aware application is no longer a difficult and time-consuming task. In consideration of the resource limitation of mobile devices, the proposed framework delegates the computationally expensive tasks to the server, while enables the context-aware mobile applications to query and subscribe high-level context information in an easy and lightweight manner.

Keywords: *Context-aware web applications, Context acquisition, Mobile devices, Ontology*

1. Introduction

Since the emergence of iPhone and Android devices, the smartphone market has shown an unprecedented prosperity. These mobile devices are equipped with increasing processing power and sensing capabilities, which allow mobile applications to sense and actively use context to provide the user with valuable information whenever and wherever is needed. For example, the user's current location information can be acquired by accessing the GPS sensor while one's hand motion can be measured by using the accelerometer sensor.

Dey puts forward the commonly used definition of context: Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves [1]. Using the context information, context-aware applications can be developed to provide personalized information and services to meet user's needs.

In the 1990s, context-aware computing started to be fully focused on and gradually became a hot research field. It helps to eliminate unnecessary user

ISSN: 2005-4270 IJDTA

cooperation and make technology as "calm" as possible [2]. And Smartphones have been regarded as the ideal platform for context-aware computing due to their features, in particular, of mobility, intelligence and personalization.

Context-aware technology has been widely used in applications such as e-learning, tour guide and healthcare. Bajnaid *et al*. [3] introduce a context-aware e-learning system to support learners developing Software Quality Assurance compliant software. Chien *et al*. [4] propose a ZigBee-ontology-based exhibit guidance and recommendation system which tracks users' visiting patterns on-line and gives real-time context-aware recommendation. Mohomed *et al*. [5] design and implement the HARMONI middleware to collect real sensor data from users for remote health monitoring. However, context-aware application development is complicated since it involves many semantic web techniques including ontology-based context modeling, context reasoning, context storage and retrieval *etc*.

To simplify the development, context-aware application frameworks have been proposed which hide the complexity of context-aware computing and provide easy-to-use application programming interface. Some frameworks [6][7] are designed especially for resource-limited mobile devices, which enable more and more context-aware applications to deploy in mobile environments, though these frameworks are usually targeted at a certain kind of mobile platform. Due to mobile platform diversity (iOS, Android, Windows Phone, *etc*.), the development and deployment of a context-aware application for different mobile devices are time-consuming and expensive, which in fact limits the large-scale application of context-aware technology in mobile heterogeneous environments.

Unlike native applications, web applications are easy to develop, upgrade and deploy, and almost all smartphones today include a web browser for supporting running them. Smartphones have become one of the most popular devices for access to web sites and web applications. Apart from that, web technologies can simplify the exchange of context information in heterogeneous environments. Another difference is that native applications can capture context information by accessing sensors available on the mobile device, but web applications running in the browser cannot, because traditional web browsers, like Chrome, Safari, *etc*., do not provide mechanism for acquiring wide variety of context information. Some articles [8-9] propose to capture user context and preference by analyzing user's web browsing habits and deliver personalized web content after context inferring, however, this approach can only acquire very limited context information.

This paper describes CaMWAF, a context-aware mobile web application framework, based on which, developers can construct cross-platform context-aware applications in an easy and fast way using HTML5, CSS3, JavaScript and simple context management APIs. Although the processing power of mobile devices has remarkably increased in recent years, they are necessarily resource-scarce compared to traditional computers. And devices in the mobile ecosystem are not equally powerful, thus the developers have to take into account the hardware configurations of each device such as battery, memory, processing capacity, *etc*. when developing applications. In consideration of the resource limitation and hardware capability difference of mobile devices, the proposed framework delegates the computationally expensive tasks, like semantic reasoning, to the server, while the mobile client is relieved from large computational burden, which enables the context-aware mobile applications to query and subscribe high-level context information in an easy and lightweight manner. Thus, the context-aware systems based on CaMWAF could be applied on a wide variety of mobile devices, even including some low-end devices.

Although context-aware applications can provide the user with personalized information and services by using context information, there is a tradeoff between having better user experience and allowing user's private information to be collected.

In order for privacy protection of user context, context information acquisition of CaMWAF is under access control.

The rest of the paper is structured as follows. Section 2 reviews the related work. In Section 3, we propose CaMWAF's architecture and describe the main components of the framework in detail. Section 4 presents the sample application built using the framework which shows the feasibility of the approach and Section 5 evaluates the performance and usability of CaMWAF. In Section 6, we conclude the paper and present future work for evolution.

2. Related work

Building context-aware applications has large development overheads due to complexities of acquiring, aggregating and inferring context information. To reduce the development burden, many context-aware frameworks have been proposed in the recent years.

Context Toolkit [10] is one of the first proposed context-aware architectures, which is a widget-based toolkit for context-aware application development. This framework proposes the concept of context widget in order to insulate applications from context acquisition concerns and presents an architecture composed of different components responsible for acquiring, aggregating and interpreting context information respectively. Approaches like SOCAM [11] or CoBrA [12], as our framework does, make use of ontology-based context model to achieve knowledge sharing and context reasoning. Yet these approaches do not fully take into account the characteristics of the mobile environment and do not regard the mobile device as the primary context provider.

With the evolution of mobile device technologies, context-aware frameworks for mobile environments have appeared in order to support the development of context-aware mobile applications, like MoBe [13], CMF [14] or Hydrogen [15]. However, different from our solution, these frameworks delegate much computational burden derived from context management to mobile phones, therefore they are not suitable for resource-scarce mobile devices.

Considering the resource-constraint of mobile phones, some frameworks have been proposed to take full advantage of the computational power of high-performance servers [6-7], but all of them are platform-specific.

To reduce the large overheads due to developing the same application for a wide variety of mobile phone models, some context-aware mobile applications like Smart Adelaide Guide [16] are developed with web technologies and executed on mobile web browsers such as Chrome, Safari, *etc*. Some solutions are proposed to collect user context by analyzing user's web access behavior [8-9], however, because of the functionality limitation of the traditional web browsers, very few context information can be acquired.

In this paper, we propose a context-aware mobile web application framework, which enables mobile web applications not only to capture various sensor data but also to be deployed on mobile devices in a simple and fast way.

3. The Proposed System

Taking into account the characteristics of mobile environments including portability, resource constraint, performance discrepancy and platform heterogeneity, the proposed framework distributes context information processing tasks between mobile client and high-performance server by delegating the computationally intensive tasks, like context reasoning, to the server, while the client is mainly responsible for providing runtime environment for context-aware web applications and collecting raw context information. The approach significantly simplifies the

development and deployment of a context-aware mobile application and achieves context knowledge sharing by adopting ontology-based context modeling.

The framework is composed of two parts as shown in Figure 1:

- Context-aware web application execution environment: It is deployed on smartphones and provides capabilities of executing context-aware web applications and capturing context information from the mobile device.
- Context-aware web service platform: It is deployed on high-performance server and takes the computationally intensive tasks derived from context management. The platform creates ontology-based context model with low-level context information received from the mobile client and infers the high-level context for subscription and retrieval according to predefined inference rules. As a result, context-aware applications can be given personalized content and services related to current context.

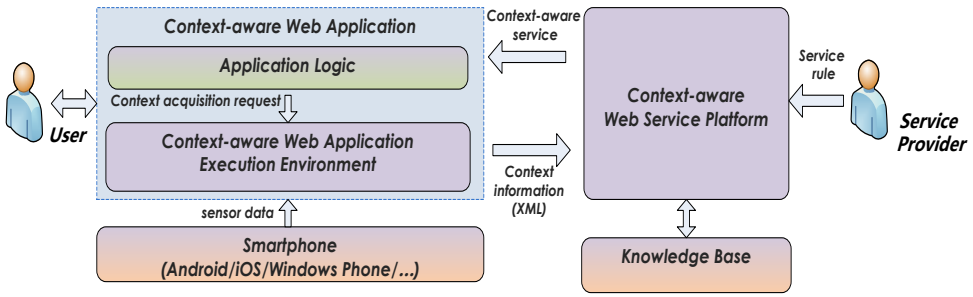


Figure 1. CaMWAF's Architecture

3.1. Context-Aware Web Application Execution Environment

The context-aware web application execution environment provides context acquisition web APIs for the context-aware web applications. To create and distribute a context-aware mobile web application, we could package the web application code (HTML, CSS and JavaScript files) together with the context-aware web application execution environment, which has small footprint and is targeted to different mobile platforms (See Figure 2).

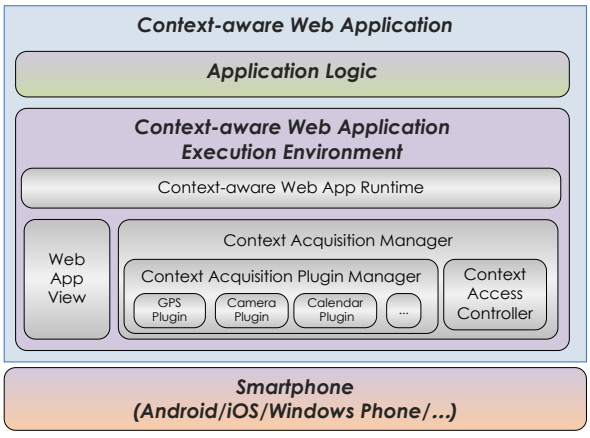


Figure 2. Architecture of the Context-Aware Web Application Execution Environment

3.1.1. Context-Aware Web App Runtime

The Context-aware Web App Runtime module carries out the concrete work of running an application, including creating the web view of the application and loading the application's web pages into it. The source code of a typical context-aware web application consists of the following files:

1. Main page file: It is coded in HTML5, JavaScript and CSS and is the entry page of the whole application.
2. Other page files: They are the page files other than the main page file and also coded in HTML5, JavaScript and CSS.
3. Resource files: They mainly include the images that the application uses.
4. Configuration file: It contains the configuration information of the application, which includes application id (the unique identifier of the application), application name, location of the entry page, context information acquisition permissions, *etc.*
5. Icon file: It is the application icon which is displayed after the application's installation.

3.1.2. Web App View

This module is used for creating views and loading application pages for displaying on screen. To enable applications to interact with the web easily, many mobile platforms, including Android, iOS and Windows Phone, implement WebView [17] component (it is called WebView in Android, UIWebView in iOS, and WebBrowser in Windows Phone, but for simplicity, we use the term WebView throughout this paper) based on the built-in web browser engine to provide APIs that can be used by applications to display web content and interact with the web content inside WebView. The Web App View module implements the functionalities including parsing, lay outing, rendering a web page by using the WebView component. Benefiting from functionality reuse, the proposed framework's mobile client is lightweight enough.

3.1.3. Context Acquisition Manager

The Context Acquisition Manager is the core module of the Context-aware Web Application Execution Environment. It manages context acquisition plug-ins and responses to context acquisition request coming from the context-aware web application (see Figure 3).

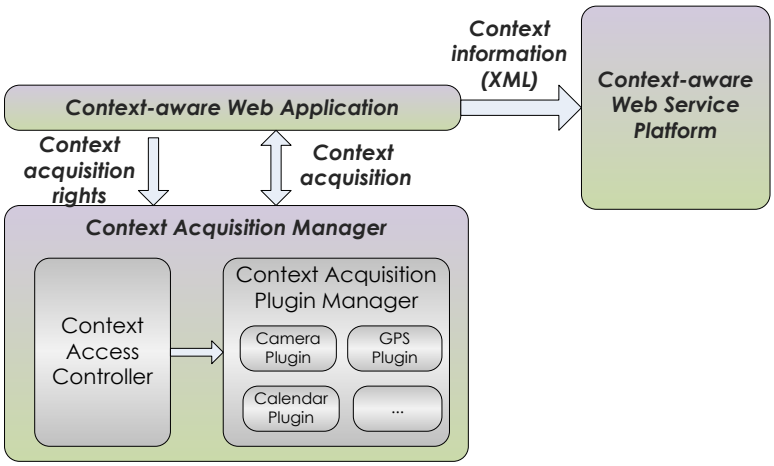


Figure 3. Architecture of the Context Acquisition Manager

The Context Access Controller sub-module is responsible for processing the context

acquisition request sent by the context-aware web application. As stated in the previous section, the source code of a context-aware web application includes a configuration file that elaborates on the context acquisition permissions of the application. The code snippet that defines the context acquisition permissions of an application is shown below.

```
<contextAcquisitionPermissions>
   <context name="Accelerometer"/>
   <context name="Camera"/>
   <context name="Contacts"/>
   <context name="Geolocation"/>
</contextAcquisitionPermissions>
```

When a context-aware web application needs to be published after its development is finished, developers require submitting it to the application store for end user's downloading. However, to ensure the validity of capturing context information from user's mobile device, the business logic code of the submitted application will be reviewed to determine whether the context acquisition permissions the developer applies for are necessary, in order to prevent malicious applications from stealing user privacy in an unauthorized manner. Meanwhile, to avoid illegal modification, the information of context acquisition permissions is encrypted. When a context-aware web application based on the proposed framework is launched at the first time, the user is notified of what kind of context information the application is going to collect and can grant permission to access specific context information among them (see Figure 4). And then, the context acquisition permission of the application will be updated accordingly.

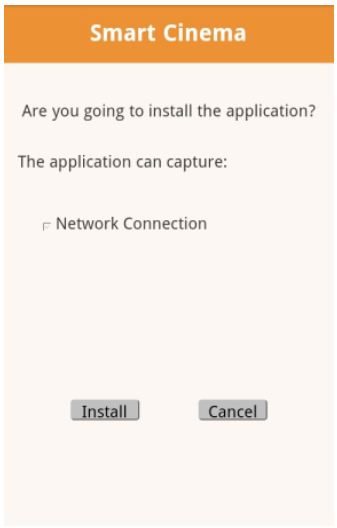


Figure 4. Configuring Context Acquisition Permissions Dynamically

When a context-aware web application sends context acquisition request, the Context Access Controller will determine whether to approve it by checking the application's context acquisition permissions. The process flow of the context acquisition access control is shown in Figure 5.

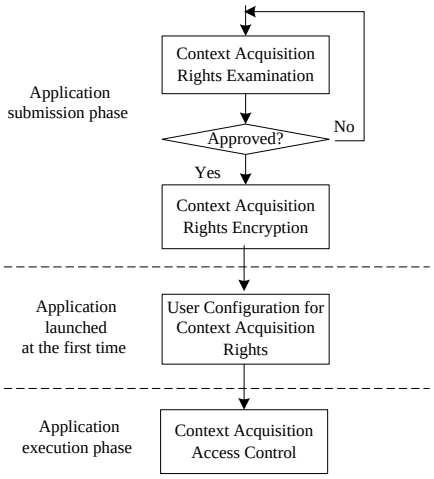


Figure 5. Flow Diagram of the Context Acquisition Access Control

Once the context acquisition request is approved, the Context Acquisition Plugin Manager can directly invoke the corresponding context acquisition plugin to acquire the raw context. Since web applications cannot capture the context information such as camera, accelerometer, contacts, *etc.* by directly invoking the native APIs exposed by the mobile platform, the framework provides JavaScript APIs which encapsulate the context acquisition functionality for accessing sensors in different types of mobile devices, and each context acquisition plugin corresponds to one kind of low-level context to be acquired. In this way, context-aware web applications are capable of collecting the same context information as context-aware native applications. For instance, the accelerometer plugin provides the following JavaScript API for capturing the current acceleration along the x, y, and z axes:

Navigator.accelerometer.watchAcceleration(successCallback:Function, errorCallback:Function, options: Object) : String;

In its parameters, the "successCallback" callback function will be invoked if the acceleration is got successfully while the "errorCallback" callback function will be called once the operation fails to get the acceleration. The "options" parameter is used for defining the frequency to retrieve the acceleration. The JavaScript function returns a watch ID which references the accelerometer's watch interval and can be used to stop watching the accelerometer.

Currently, the context acquisition plugins we have implemented include Accelerometer, Battery, BarcodeScanner, Calendar, Camera, Capture, Compass, Contacts, Geolocation, NetworkConnection, DeviceInfo, *etc.*, which are used to collect raw context data.

As mobile devices are equipped with more and more sensors, the Context Acquisition Plugin Manager provides the extension mechanism for easily adding new context acquisition functionality plugins. When CaMWAF is initialized, the Context Acquisition Plugin Manager loads the required context acquisition plugins according to the content of the plugins configuration file. The code snippet of the plugins configuration is shown below:

```
<plugins>
    <plugin name="Accelerometer" value="com.extension.AccelerometerContext"/>
    <plugin name="Camera" value="com.extension.CameraContext"/>
    <plugin name="Contacts" value="com.extension.ContactContext"/>
    <plugin name=" DeviceInfo" value="com.extension.DeviceContext"/>
    <plugin name="Geolocation" value="com.extension.GeolocationContext"/>
</plugins>
```

After finishing the context acquisition operation, the raw context data will be encapsulated in XML format and sent to the server via RESTful [18] API.

In addition, context-aware applications may need to collect other context information according to their requirements. For example, it is necessary for the Smart Advertisement application to acquire user context information such as hobbies, profession, age, *etc.*

3.2. Context-Aware Web Service Platform

The Context-aware Web Service Platform is responsible for converting the low-level context information received from the mobile client to ontology-based context model, inferring the high-level context according to predefined inference rules, and offering personalized content or services for subscription and retrieval by context-aware mobile web applications. Its architecture is shown in Figure 6.

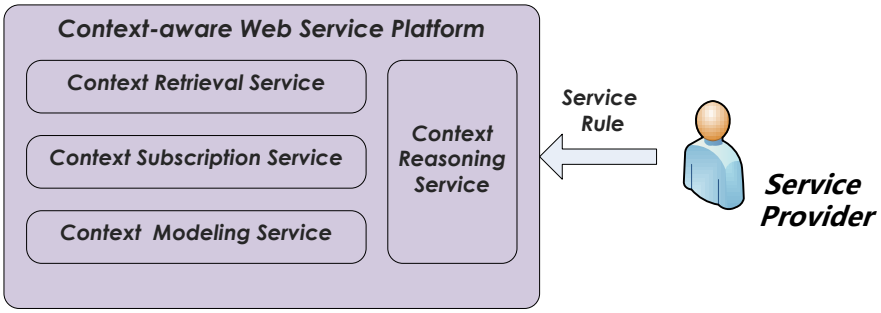


Figure 6. Architecture of the Context-Aware Web Service Platform

The responsibility of the Context Modeling Service module is to convert the raw context information to context representation model. Due to the functional difference among sensors, the collected raw sensor data has different representation. How to create a common model which effectively organizes the various contexts is a problem to be resolved. In CaMWAF, an ontology-based context model is used which offers semantic interoperability and supports knowledge reuse and context reasoning [19-20]. Traditional SOAP-based web services is heavyweight, thus it is not suitable for resource-limited mobile devices. RESTful web services is a newer solution for lightweight web service implementation. It is based on resource-oriented access where RESTful web services are basically access paths to web resources. Therefore, the context modeling service that CaMWAF provides is accessed using the lightweight RESTful web services. Jena Semantic Web Toolkit [21] is used to carry out the concrete context modeling and storage work. Figure 7 demonstrates the work flow of creating and storing context ontology instance using Jena.

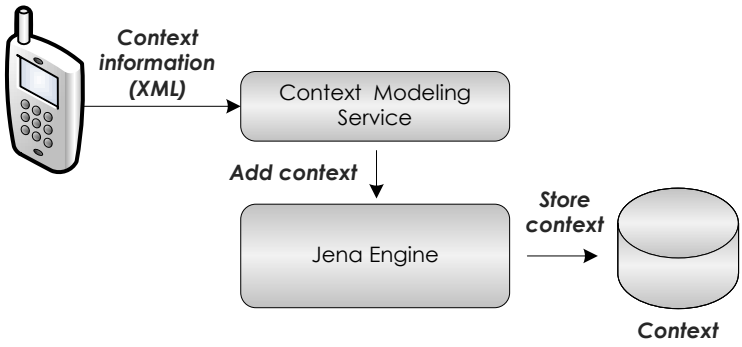


Figure 7. Work Flow of Adding and Storing Context Ontology Using Jena

Generally, the context-aware applications are more interested in the high-level context information than the low-level ones. By performing semantic reasoning over the context ontology, high level implicit context can be derived from low level explicit context. For instance, the user's current physical activity, such as walking, can be inferred from the device's acceleration data collected by the accelerometer sensor. Apart from that, consistency of the context can be checked and inconsistencies can be eliminated. In the proposed framework, by using Jena, the Context Reasoning Service module makes rule-based reasoning over the context ontology created by the Context Modeling Service module. The rule-based reasoning process may be triggered according to a predefined rule set whenever the context changes. The context-aware application can update or extend the inference rules as required.

The Context Subscription Service module provides capability of subscribing context information. As a context consumer, the context-aware application is enabled to receive asynchronous context information when given conditions are satisfied by the current context. To achieve this goal, context-aware applications need to register the context subscription information which includes conditions and actions to be executed when those conditions are satisfied. In this way, the Context-aware Web Service Platform will deliver appropriate context-aware service to the mobile application according to the service rules the service provider provides. To take Smart Meeting, the context-aware mobile web application, as an example, one of its service rules is shown below:

Table 1. Service Rule Example of the Smart Meeting Application

Conditions	Actions
(?participant smartMeeting:hasLocation smartMeeting:meetingRoom)	mute phone

As shown in Table 1, once the system finds that the condition is satisfied, which means the participant is currently in the meeting room, it will automatically send the action to the context-aware mobile application, which will then set the user's mobile phone to silent mode.

The Context Retrieval Service module is responsible for processing synchronous context information request from the mobile application. The concrete context query work is carried out by Jena Semantic Web Toolkit as well. The application can retrieve the context information by providing SPARQL [22] queries. The work flow of retrieving context information synchronously is shown in Figure 8.

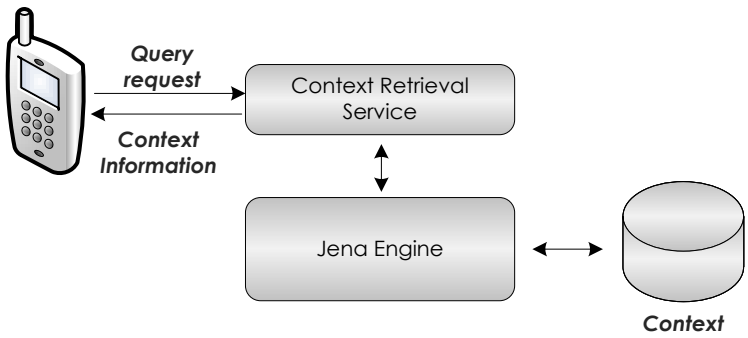


Figure 8. The Work Flow of Retrieving Context Information

4. Sample Application

To validate the approach, Smart Cinema, a context-aware web application built with the proposed framework, is implemented and deployed on the Android device. The

application provides the user with the recommended movies shown at the cinema based on user's context information.

The application needs to collect context information such as user's current network connection, favorite movie category, *etc.* To access the physical sensor, we add the following context acquisition permission to the configuration file of the application:

```
<contextAcquisitionPermissions>
    <context name="NetworkConnection"/>
</contextAcquisitionPermissions>
```

The reason why network connection information is captured is that the application will push the user more movie information, especially the movie pictures, if it find that the user is currently using WiFi connection instead of cellular data connection, as he has to pay for the data used over the mobile operator's cellular network. The network connection context, as well as application-specific context, like the user's favorite movie category, are encapsulated in XML format and sent to the server through the RESTful API for context processing.

Meanwhile, the application development process also includes creating application domain ontology, defining context inferring rules and context-aware service rules, and building application's server-side program. Figure 9 shows the ontology-based context model of Smart Cinema.

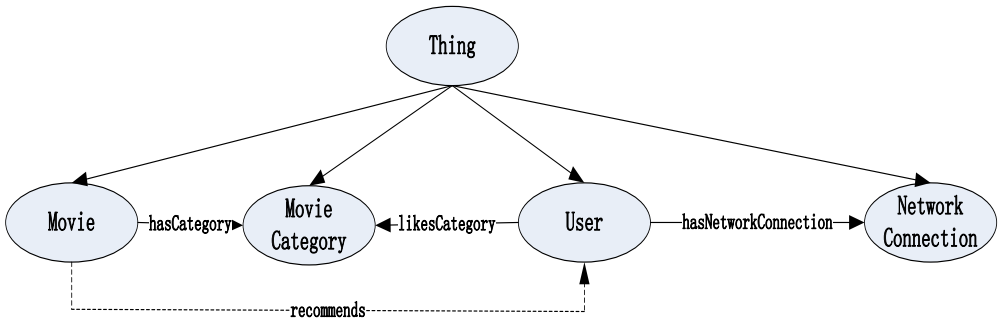


Figure 9. The Context Model of Smart Cinema

The ontology models information about movie, user, movie category, as well as network connection. And the highly recommended movies can be inferred if their category is the same as user's favorite movie category and they have high guest ratings. The inferring rule is shown in Listing 1.

Listing 1. Inferring Rule of Detecting Highly Recommended Movies

```
(?movie smartCinema:hasUserRating ?userRating)
greaterThan(?userRating, 8)
(?user smartCinema:likesCategory ?movieCategory)
(?movie smartCinema:hasCategory ?movieCategory)
->
(?movie smartCinema:recommends ?user)
```

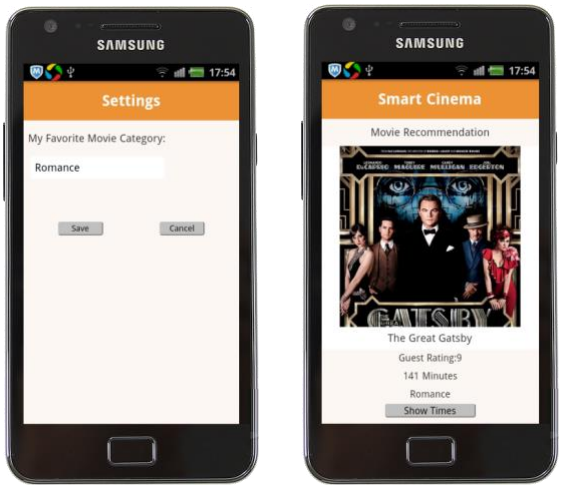
In order to deliver the context-aware services to the mobile application automatically, the application registers its context subscription information in the service rules.

Table 2. One Service Rule of the Smart Cinema Application

Conditions	Actions
(?movie smartCinema:recommends ?user) ˆ(?user smartCinema:hasNetworkConnection smartCinema:WiFi)	push recommended movies with pictures

As shown in Table 2, once the application is running and the Context-aware Web Service Platform infers the highly rated movies which belong to the user's favorite movie category by using the predefined context inference rules, the server application will push the user the movies with the highest guest ratings by executing the predefined actions.

Figure 10(a) demonstrates the application's settings UI where the user can set his movie preference information like his favorite movie category.

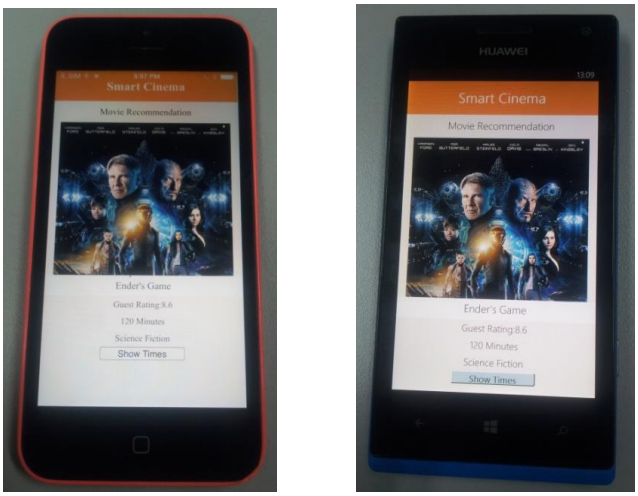


(a) (b)

Figure 10. (a) Settings UI of the Smart Cinema Application (b) The Movie Recommendation Information of the Smart Cinema Application

Figure 10(b) shows the recommended movie information which includes movie name, poster, guest rating, length, category and show times as well.

In addition, as the proposed framework is cross-platform, Smart Cinema can also run without code modification on the iOS and Windows Phone 8 devices as shown in Figure 11.



(a) (b)

Figure 11. (a) The Movie Recommendation UI of Smart Cinema Running on iPhone. (b) The Movie Recommendation UI of Smart Cinema Running on Windows Phone 8 Device

5. Evaluation

In this section, we will evaluate performance and usability of CaMWAF by using the Smart Cinema web application, which has been developed based on it, and a context-aware native application with the same functionality. To obtain user evaluation, we invite developers to participate in context-aware application development with the proposed framework.

5.1. Performance Evaluation

In order to assess the performance of CaMWAF, tests have been carried out to evaluate the most commonly used operations including context retrieval and context subscription. The two applications that are used for the tests are:

- The Smart Cinema application that is developed with native technology: Specifically, its mobile client is an Android application which is allowed to collect context information by accessing the sensors directly, while it makes use of CaMWAF's service platform for context management.
- The Smart Cinema application that is developed with web technology: It is developed fully based on CaMWAF and has the same functionality as the one developed based on Android platform.

The mobile device where we run the two applications is a Samsung Galaxy S II smartphone that has the following features: 1.2 GHz Dual-Core CPU, 1GB RAM, 4G ROM, Android OS 4.2. The server is DELL PowerEdge R710 with features including Intel Xeon 2.26GHz Quad-Core CPU, 12G RAM and Ubuntu 12.04. The mobile client is connected to the server through a WiFi network.

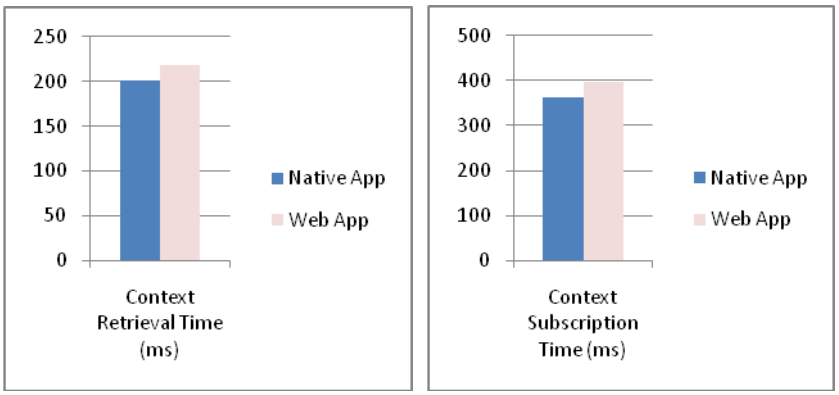
The test scenarios include:

(A) Context retrieval: the mobile application retrieves context information of the movies whose category are romance. The SPARQL query is shown in Listing 2. The test result is shown in Figure 12 (a).

Listing 2. The SPARQL Query for Retrieving Romance Movies

```
PREFIX smartCinema: <http://www.semanticweb.org/administrator/ontologies/smartCinema.owl#>
SELECT ?name
WHERE {
        ?movie smartCinema:hasName ?name .
        ?movie smartCinema:hasCategory smartCinema:Romance
        }
```

(B) Context subscription: the mobile application updates user's favorite movie category. The context change immediately triggers context reasoning and then the Context-aware Web Service Platform delivers personalized movie recommendation to the mobile client. The context subscription time is shown in Figure 12 (b). The start time of the subscription is when the user context is updated, while the end time is when the mobile application receives the asynchronous context information.



(a) (b)

Figure 12. (a) Context Retrieval Time (ms) Comparison between the Evaluated Applications (b) Context Subscription Time (ms) Comparison between the Evaluated Applications

The context retrieval performance described in Figure 12 (a) gives similar values in the evaluated applications. This is because they use the same RESTful web service to perform context queries. The context subscription performance is shown in Figure 12 (b). As the time-consuming tasks like context retrieval, semantic reasoning, *etc.* are all carried out at the server side and the mobile client is only responsible for updating the context in this scenario, both of the two applications show good performance.

From the obtained results, we can conclude that the proposed framework has good performance and the evaluated web application built with CaMWAF is as efficient as the traditional context-aware native application.

5.2. Usability Evaluation

To evaluate the usability of the proposed framework, we invite experienced application developers to participate in the context-aware application development by using CaMWAF.

All of the participants state that CaMWAF is much easier to use than expected. Also, they state that the proposed framework provides high extensibility as the functionality of one context-aware application can be extended in an easy and fast way.

Table 3. Comparison with Native Framework

Factor	CaMWAF	Native framework
Application deployment on heterogeneous mobile devices	Good	Poor
Easy to learn	Good	Moderate
Application development efficiency	Good	Moderate

As shown in Table 3, CaMWAF is much better than the native framework in supporting heterogeneous mobile devices. Currently, CaMWAF supports the major mobile platforms including Android, iOS and Windows Phone. CaMWAF is web-based, and HTML5 mobile web development is easier to learn than native mobile application development since HTML is a very familiar and simple language. Meanwhile, application development efficiency of CaMWAF is better. With CaMWAF, context-aware web application developers only need to develop the application once and then deploy it on different types of mobile devices without source code modification, which avoids building applications written in different native languages for different platforms, and

thus greatly reduces the overhead of developing a context-aware application and saves development costs.

6. Conclusion

In this paper, we propose a web-based framework for context-aware mobile web applications, which makes it possible to efficiently develop context-aware applications and deploy them on heterogeneous mobile devices. Due to resource-constraint of mobile environments and large resource consumption derived from context information processing, the proposed framework reasonably distributes context information processing tasks between mobile client and high-performance server by delegating the computationally expensive tasks, like semantic reasoning, to the server side, while the mobile client is relieved from large computational burden and enables the context-aware mobile applications to query and subscribe high-level context information simply and rapidly. In this way, the computational power of the server is fully taken advantage of and the context management overhead of the mobile client is reduced as much as possible.

There are various mobile platforms today and building a context-aware application for each platform is very expensive if written in each native language. However, the proposed framework is based on HTML5 web technology, which not only simplifies context-aware application development, but also enables cross-platform deployment of context-aware applications.

The privacy protection of user context is a key problem to be resolved by context-aware systems. Through context acquisition access control, the framework prevents malicious applications from stealing user privacy in an unauthorized way. However, security of user's context information collected validly should also be guaranteed. Therefore, the framework needs to enhance the context security management in future.

The test result shows that the evaluated web application built with CaMWAF is as efficient as the traditional context-aware native application. However, compared to the traditional context-aware native application framework, CaMWAF is much easier to use for developing and deploying cross-platform context-aware applications, and thus can facilitate the large-scale application of context-aware technology in mobile heterogeneous environments.

Our future work will mainly focus on the following areas:

- Improving the user context privacy protection mechanism by providing secure retrieval and storage of context information.
- Implementing new context acquisition plugins for other types of context information.
- Providing useful and powerful context-aware web application development tools to reduce the overhead of developing a context-aware application and make the development as simple as possible.

Acknowledgement

The authors wish to acknowledge the support of the Development of Smartphone Supporting Web Technologies project (ID: 2014ZX03002001) of Important National Science & Technology Specific Projects of Chinese Government for funding the research work presented in this paper.

References

[1] A. K. Dey, “Understanding and using context”, Personal and Ubiquitous Computing, vol. 5, **(2001)**, pp. 4-7.
[2] G. Chen and D. Kotz, “A survey of context-aware mobile computing research”, Technical Report No. TR2000-381, Department of Computer Science, Dartmouth College, Hanover, NH, USA, **(2000)**.

[3] N. Bajnaid, R. Benlamri and B. Cogan, "Context-aware SQA e-learning system", In Proceedings of the 2011 6th International Conference on Digital Information Management, **(2011)**, pp. 327-331.
[4] H.-Y. Chien, S.-K. Chen, C.-Y. Lin, J.-L. Yan, W.-C. Liao, H.-Y. Chu, K.-J. Chen, B.-F. Lai and Y.-T. Chen, "Design and implementation of ZigBee-ontology-based exhibit guidance and recommendation system", International Journal of Distributed Sensor Networks, vol. 2013, **(2013)**.
[5] I. Mohomed, A. Misra, M. Ebling, and W. Jerome, "HARMONI: Context-aware filtering of sensor data for continuous remote health monitoring", In Proceedings of the 6th Annual IEEE International Conference on Pervasive Computing and Communications, **(2008)**, pp. 248-251.
[6] S. De and K. Moessner, "A framework for mobile, context-aware applications", In Proceedings of the 16th International Conference on Telecommunications, **(2009)**, pp. 232-237.
[7] D. Martín, C. Lamsfus and A. Alzua, "Mobile context data management framework", In Proceedings of the 2011 5th FTRA International Conference on Multimedia and Ubiquitous Engineering, **(2011)**, pp. 73-78.
[8] D. Wolff, M. Schaaf, S.G. Grivas and U. Leimstoll, "Context-aware website personalization", Lecture Notes in Computer Science, vol. 6882, **(2011)**, pp. 51-62.
[9] X. Zhang, Z. Yu, J. Tian, Z. Wang and B. Guo, "Context-aware mobile web browsing based on html5", In Proceedings of IEEE 9th International Conference on Ubiquitous Intelligence and Computing and IEEE 9th International Conference on Autonomic and Trusted Computing, **(2012)**, pp. 945-950.
[10] A. K. Dey, G. D. Abowd and D. Salber, "A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications", Human Computer Interaction, vol. 16, **(2001)**, pp. 97-166.
[11] T. Gu, H. K. Pung and D. Q. Zhang, "A Middleware for Building Context-Aware Mobile Services", In Proceedings of IEEE Vehicular Technology Conference, **(2004)**, pp. 2656-2660.
[12] H. Chen, T. Finin and A. Joshi, "Semantic web in the context broker architecture", In Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications, **(2004)**, pp. 277-286.
[13] P. Coppola, V. Della Mea, L. Di Gaspero, S. Mizzaro, I. Scagnetto, A. Selva, L. Vassena and P. Riziò, "MoBe: Context-aware Mobile Applications on Mobile Devices for Mobile Users", In Proceedings of the International Workshop on Exploiting Context Histories in Smart Environments, **(2005)**.
[14] P. Korpipää, J. Mantyjarvi, J. Kela, H. Keranen and E. Malm, "Managing context information in mobile devices", IEEE Pervasive Computing, vol. 2, **(2003)**, pp. 42-51.
[15] T. Hofer, W. Schwinger, M. Pichler, G. Leonhartsberger and J. Altmann, "Context-awareness on mobile devices – the hydrogen approach", In Proceedings of the 36th Annual Hawaii International Conference on System Sciences, **(2002)**, pp. 292-302.
[16] K. Liao, Q. Z. Sheng, J. Yu, and H. S. Wong, "Smart Adelaide guide: A context-aware web application", In Proceedings of the 11th International Conference on Information Integration and Web-based Applications and Services, **(2009)**, pp. 681-687.
[17] http://developer.android.com/reference/android/webkit/WebView.html.
[18] L. Richardson and S. Ruby, "RESTful Web Services", O' Reilly Media: Sebastopol, CA, **(2007)**.
[19] X. H. Wang, D. Q. Zhang, T. Gu and H. K. Pung, "Ontology based context modeling and reasoning using OWL", In Proceedings of the second IEEE annual conference on pervasive computing and communications, **(2004)**, pp. 18-22.
[20] T. Gu, X. H. Wang, H. K. Pung and D. Q. Zhang, "An ontology-based context model in intelligent environments", In Proceedings of communication networks and distributed systems modeling and simulation conference, **(2004)**, pp. 270-275.
[21] B. McBride, "Jena: A semantic web toolkit", IEEE Internet Computing, vol. 6, **(2002)**, pp. 55-59.
[22] http://www.w3.org/TR/rdf-sparql-query/.

Authors

Jianchao Luo, He is a Ph.D. candidate in Computer Science at University of Electronic Science and Technology of China. He has been a lecturer in School of Computer Science and Engineering, University of Electronic Science and Technology of China. His research interests include mobile computing, context-aware systems and e-commerce security.

Hao Feng, He received his B.S. and M.S. degrees in Computer Science and Engineering from University of Electronic Science and Technology of China, in 2000 and 2004, respectively. He has been an associate professor in School of Computer Science and Engineering, Guilin University of Electronic Technology. His research interests include sensor network, context-aware system and mobile computing.

