International Journal of Bio-Science and Bio-Technology
Vol.7, No.1 (2015), pp.207-218
http://dx.doi.org/10.14257/ijbsbt.2015.7.1.22

# U-healthcare Enterprise Frameworks for Mobile Applications

Haeng-Kon Kim

*Dept. of Computer Information & Communication Engineering*
*Catholic Univ. of Daegu, Korea*

***Abstract***

*The advances in mobile devices, frameworks and u-computing have given rise to a vast range of new services for RFID (Radio Frequency Identification) technology. In this paper, we propose a U-healthcare Enterprise Application Framework for interchanging the information on mobile devices in the RFID-based distributed computing environment. We describe the requirements of u-healthcare Enterprise Application Framework, mobile applications development architectures and EPC global network which is RFID-based network environment. It include an architectures and scenarios how to construct the u-healthcare enterprise applications using the proposed Enterprise Application Framework by changing the contents of the Application Layer.*

***Keywords:*** *Mobile Development Environments, Enterprise Architecture, EPC RFID, U-Computing. Middleware, Enterprise Application*

## 1. Introduction

Researches and developments for ubiquitous computing environment which is human centered future computing environment are widely proceeded. One of them is the research and development of RFID (Radio Frequency Identification) technology [1-2]. The EPC (Electronic Product Code) Network, which was developed by the Auto-ID Center and now managed by EPC global Inc., was designed and implemented to enable all objects in the world to be linked via the Internet and mobile [3]. The functions of the EPC Network using RFID tags are as follows: recognize, identify all objects, track, trace, monitor, trigger events and actions on those objects, and offer real-time view of assets and inventories throughout the global supply chain [4]. Mobile application development by which application software is developed for handheld devices, such as personal digital assistants, enterprise digital assistants or mobile phones. These applications can be pre-installed on phones during manufacturing platforms, or delivered as web applications using server-side or client-side processing (*e.g.* JavaScript) to provide an "application-like" experience within a Web browser. Application software developers also have to consider a lengthy array of screen sizes, hardware specifications and configurations because of intense competition in mobile software and changes within each of the platforms [5-6].

In this paper, we propose a U-healthcare Enterprise Application Framework for interchanging the information on mobile devices in the RFID-based distributed computing environment. We describe the requirements of u-healthcare Enterprise Application Framework, mobile applications development architectures and EPC global network which is RFID-based network environment. It include an architectures and scenarios how to construct the u-healthcare enterprise applications using the proposed Enterprise Application Framework by changing the contents of the Application Layer.

ISSN: 2233-7849 IJBSBT
Copyright ⓒ 2015 SERSC

### 2.1. Mobil U-healthcare Development Model [6]

There is no standard model or rule to design and develop mobile u-healthcare application. In order to support the medical specialists such as doctor, physician and therapist for developing personalized mobile u-healthcare applications, an application model for the mobile u-healthcare applications should be defined prior to the implementation of the workbench. The suggest application model consists of five phases. Sensing phase collects various vital signs of patients from mobile bio-sensors such as thermometer, ECG (electrocardiogram), PPG (photoplethysmography), fat measurer. The sensors are usually connected with mobile gateways through wired or wireless connections. Questionnaire phase collects health data that cannot be obtained by sensors. Many kinds of questionnaire are used according to the types of diseases. Data processing phase is to process the data collected by the first and second phases. The collected data is analyzed to decide the status of patient's diseases. Sometimes training data is used for the decision. Disease treatment phase provides or suggests appropriate treatment programs based on the analyzed results of the previous phase. For example, deep breathing and meditation can be recommended for the patients suffering from mental stresses. The final feedback phase is for getting user's feedback like the satisfaction of the service, or the improvements of the symptoms. Our u-health application development platform is implemented in the form of a workbench so that the medical specialists develop their own u-health services according to the guideline derived from the application model in the above. The following activities of medical specialists are supported in the workbench. Once the type of diseases for service is decided, the medical specialists design the types of vital signs and choose sensors to be used in collecting health data. Then the medical specialists design questionnaires to gather symptom or other environment information. Under the assumption that the health data is successfully obtained in manageable form through bio-sensors and questionnaires, the medical specialists design the analysis method for the input health data. A simple health index may be used or some sophisticated analysis method such as expert system or some learning method may be designed or adopted. The degree of freedom of this phase is so high and thus this part is so platform dependent. Our application development platform also provides a unique and general matrix-based two-category classification method for this phase. The medical specialists may design a training based disease group identification method using the method. The spectrum of disease treatments is as wide as the analysis methods of health data. Each medical specialist may develop one's own treatment method or program. Under the assumption that numerous u-health treatment methods for a specific disease are available on the Internet, our workbench provides a treatment recommendation or ranking method based on user's vital-sign and symptom combination information. Finally, the medical specialists design a feedback mechanism for gathering the responses on the treatments from the users. Sometimes the feedback data is accumulated to improve the analysis method or post-processing of services. In MUSS [6] application, MUSS platform consists of three layers as depicted in Figure 2. The component layer contains service components which are used for the composition of the services in developing a mobile u-healthcare application. The process layer includes healthcare processes which specify the logic of mobile u-healthcare application, and actual personalized applications such as stress, obesity, diabetes applications, and so on are placed on the application layer.

 Copyright ⓒ 2015 SERSC

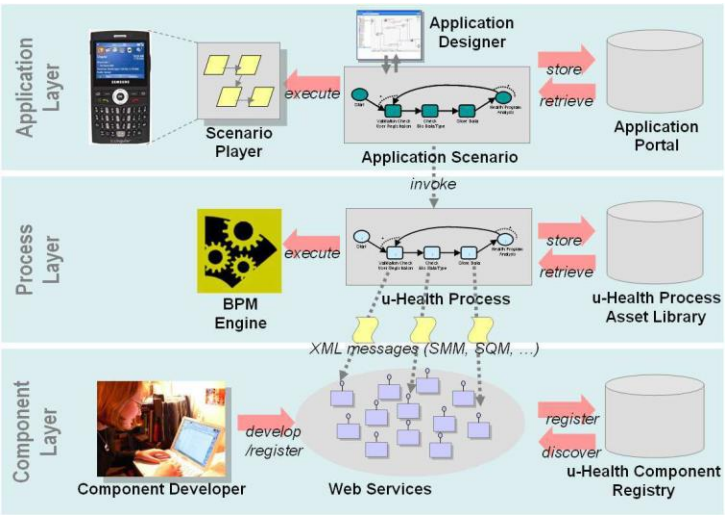


**Figure 1. The MUSS [6] Platform Architecture**

## 2.2. EPC Global Network

The EPC Network, which was developed by the Auto-ID Center and now managed by EPC global Inc., was designed and implemented to enable all objects in the world to be linked via the Internet [7]. The architecture of EPC global Network is shown in Figure 2.

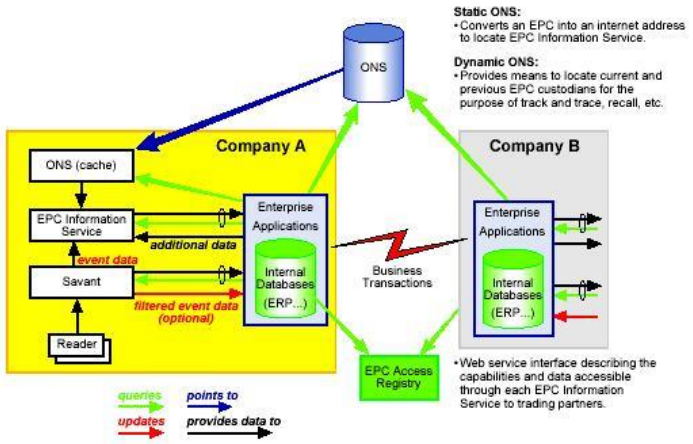


**Figure 2. The Architecture of EPC Global Network [*source: Savant spec*]**

EPC (Electronic Product Code) code is the numeric data transmitted by a tag which is the next generation of the UPC (Universal Product Code), that is bar code. Unlike the UPC, the EPC is designed to operate not only wirelessly, but to uniquely identify each individual object [6]. Typical EPC Network consists of three parts: Savant, EPCIS, and ONS. SAVANT is middleware between RFID Reader(s) and application which passes requests from applications to reader(s) and receives unique tag identifiers and possibly other data from sensors, and passes that information to the applications [5]. Event management in Savants is main function such as filtering. Because readers may read data coming from multiple tags many times, it is inevitable to eliminate redundant or unnecessary information. EPCIS (Electronic

Copyright ⓒ 2015 SERSC

Product Code Information Service) communicates with databases and stores the tag data when it is read. ONS (Object Name Service) identifies the location of the server hosting the appropriate information needed by the application [5].

### 2.3. Mobile Applications Development Requirements

As part of the development process, Mobile User Interface (UI) Design is also an essential in the creation of mobile apps. Mobile UI considers constraints & contexts, screen, input and mobility as outlines for design. The user is often the focus of interaction with their device, and the interface entails components of both hardware and software. User input allows for the users to manipulate a system, and device's output allows the system to indicate the effects of the users' manipulation. Mobile UI design constraints include limited attention and form factors, such as a mobile device's screen size for a user's hand(s). Mobile UI contexts signal cues from user activity, such as location and scheduling that can be shown from user interactions within a mobile application. Overall, mobile UI design's goal is primarily for an understandable, user-friendly interface. The UI of mobile apps should: consider users' limited attention, minimize keystrokes, and be task-oriented with a minimum set of functions. This functionality is supported by Mobile enterprise application platforms or Integrated development environments (IDEs). Mobile UIs, or front-ends, rely on mobile back-ends to support access to enterprise systems. The mobile back-end facilitates data routing, security, authentication, authorization, working off-line, and service orchestration. This functionality is supported by a mix of middleware components including mobile app servers, Mobile Backend as a service (MBaaS), and SOA infrastructure. The following software platforms will run on hardware platforms from a number of different manufacturers:

- Java ME This platform generally produces portable applications, although sometimes device-specific libraries exist (commonly used for games), making them non-portable. It is often used to provide simple applications on feature phones. Applications (including their data) cannot be larger than around 1 MB if they are to run on most phones. They must also be cryptographically signed in order to use APIs such as the file system access API. This is relatively expensive and is rarely done, even for commercial applications. Java ME runs atop a *Virtual Machine* (called the KVM) which allows reasonable, but not complete, access to the functionality of the underlying phone. The JSR process serves to incrementally increase the functionality that can be made available to Java ME, while also providing Carriers and OEMs the ability to prevent access, or limit access to provisioned software.
- Symbian platform Designed from the start for mobile devices, the Symbian platform is a real time, multi-tasking OS specifically architected to run well on resource-constrained systems, maximizing performance and battery life whilst minimizing memory usage. The Symbian Foundation maintains the code for the open source software platform based on Symbian OS and software assets contributed by Nokia, NTT DOCOMO, and Sony Ericsson, including the S60 and MOAP(S) user interfaces. The platform is fully open source, mostly supplied under the Eclipse Public License. Over 300 million Symbian OS-based units have been shipped and Symbian holds more than a 50% market share globally.
- Android is a Linux-based platform from the Open Handset Alliance, whose 34 members include Google, HTC, Motorola, Qualcomm, and T-Mobile. It is supported by over 34 major software, hardware and telecoms companies. The Linux kernel is used as a hardware abstraction layer (HAL). Application programming is primarily done in Java. The Android specific Java SDK is required for development although any

 Copyright ⓒ 2015 SERSC

Java IDE may be used. Performance critical code can be written in C, C++ or other native code languages using the Android Native Development Kit (NDK).

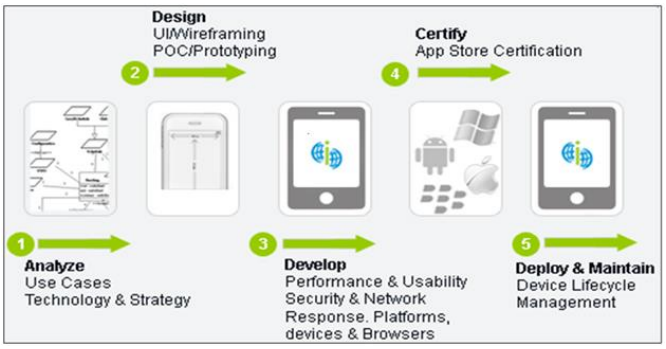


**Figure 3. Mobile Applications Development Architectures**

## 3. U-healthcare Enterprise Application Framework

### 3.1. Architecture of U-healthcare Enterprise Application Framework

Architecture of U-healthcare Enterprise Application Framework using CBD (Component-Based Development) and mobile services is shown in Figure 4. Application Architecture Layer consists of domain dependent components in which domain specific modules such Actors and Artifacts and domain applications knowledge may be situated.

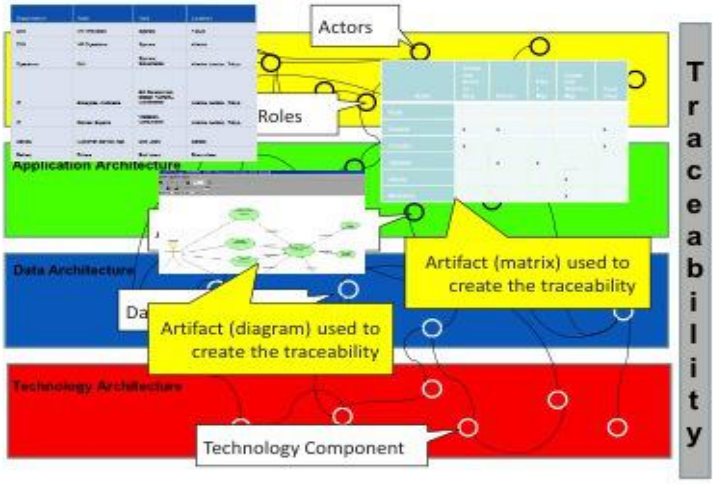


**Figure 4. Architecture of U-healthcare Enterprise Application Framework**

Framework Layer consists of domain independent components in which domain neutral modules such as Security Services, Messaging Modules, Transaction Manager, and so on, may be situated. These modules need not to be modified even if the domain is changed. This is one advantage of CBD (Component-Based Development). RFID Middleware Layer passes requests from applications to reader(s) and receives unique tag identifiers, and passes that information to the applications. RFID readers consist of an antenna and control unit which

Copyright ⓒ 2015 SERSC

monitors encoding and decoding, data check and save, tag and communication with host, and so on.

**- Application Layer**

Application Layer consists of domain dependent components in which domain specific modules may be situated.

**- UDDI Access Module**

UDDI is a collection of shared directory or protocol that we can register Web Services and can search for the registered service by the real time.

**- WSDL Module**

WSDL is an interface language that informs outside users how to use functions offered by the Web Services. WSDL corresponds to IDL (Interface Definition Language) of CORBA.

**- Repository Module**

Repository Module is a database that stores WSDL document whose contents are domain specific.

**- Framework Layer**

Framework Layer consists of domain independent components in which domain neutral modules may be situated. These modules need not to be modified even if the domain is changed.

**- Authentication Module**

Authentication module guarantees that only the users who are given rights can access to the application, and the module is divided by authentication and authorization. Authentication is a technique that transmits user's identity, and authorization is a technique that judge availability whether it comforts to the required access level. We will install an authentication server in the center only because there are difficulties in the key management if all servers try to authenticate clients directly in the distributed environment. Also we introduce the technique of single sign-on that allows a client to connect repeatedly to the several servers by the password authentication at one time.

**- Digital Signature Module**

Digital Signature Module handles digital signature [10] which is used to secure the writer's identity by the electronic way, and has the functions of integrity and authentication at the same time. Also it can be used for the purpose of non-repudiation. This module can issue digital signature of XML document, and can provide the function of integrity with the document.

**- Encryption Module**

The practice of hiding messages so that they cannot be read by anyone other than the intended recipient. There are two types of encrypting methods : symmetric and asymmetric. In the symmetric method, sender and recipient share a common key. In the asymmetric method, there are two keys – a public & a private key. The public-key, which may be known by anybody, and can be used to encrypt messages, and verify signatures a private-key, known only to the recipient, used to decrypt messages, and sign signatures. This is asymmetric because those who encrypt messages or verify signatures cannot decrypt messages or create signatures [8]. Our proposed Encryption Module supports both symmetric cipher and asymmetric cipher, and also provides the function of XML Encryption [9].

**- Reliable Messaging Module**

Reliable Messaging Module securely guarantees reliable message between sender and recipient. Reliable message transmission is based on the session technology, which has ability to provide numbering with the messages in the same session, has acknowledgment function to prove the receipt of message, and has verification function for the specific message.

 Copyright ⓒ 2015 SERSC

**- Transaction Module**

The usual distributed transaction modules try to lock against transaction failure. Web Services, however, can not do that because the category and scale of transactions are so great [2]. The compensating transaction takes place whenever transaction failure is occurred.

**- Messaging Module**

Messaging Module takes charge of creating SOAP messages which will be transmitted. If necessary, name space, header, body, encryption, digital signature, authentication information, and so on may be added.

**- Web Server Module**

Web Server Module is in charge of transmitting SOAP messages. SOAP message may be applied to the SMTP protocol as well as HTTP protocol. However, SOAP message utilizes only HTTP since it is very difficult to control diverse protocols appropriately in the viewpoint of the specific protocols.

**- Logging Module**

Logging Module stores information on the transaction failure, message transmission failure, system error, and so on.

### 3.2. Application Scenario for U-healthcare Applications

Figure 5 is a typical example of Application Scenario for U-healthcare applications based on RFID Application using AAF which is a light-version of EAF (Enterprise Application Framework), suggested in this paper. AAF is constructed on the basis of ALE engine. EPC global network shows a new standard, called Client Container *(CC)*, which is developed from the concept of a middleware. The role of the CC is to provide a means to process the event data which were collected by the RFID reader and deliver them to the higher-level applications [4, 11].

Application Scenario for U-healthcare applications is dynamic modeling architecture of a framework whose purpose is to allow the user to develop mobile application with ease. After developing CC, we are trying to extend it to Data Container(DC) framework by replacing BP (Business Process) bridge mapping component of CC and DC like a business enterprise framework) component which is under development by another team of our research group.

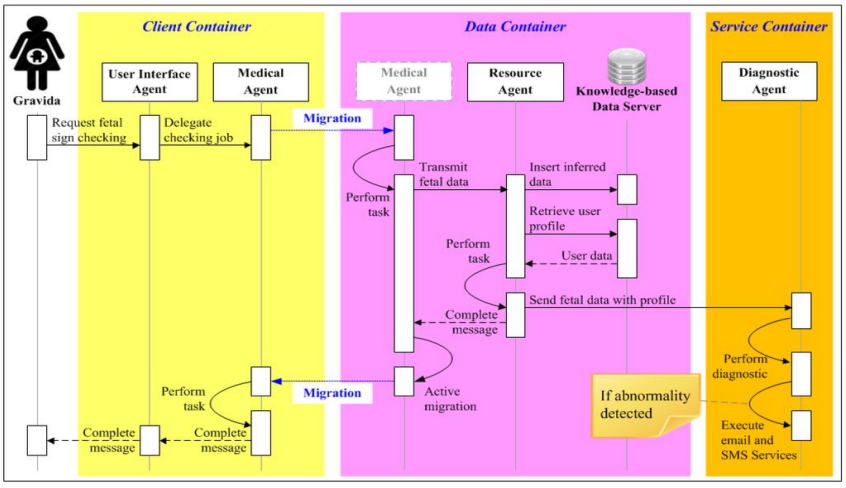


**Figure 5. Application Scenario for U-healthcare Applications**

Copyright ⓒ 2015 SERSC

**Scenario 1:** Client defines an interested Client Container using patient fatal Client API, and produces CC Spec requesting for u-healthcare information event as in Figure 5.

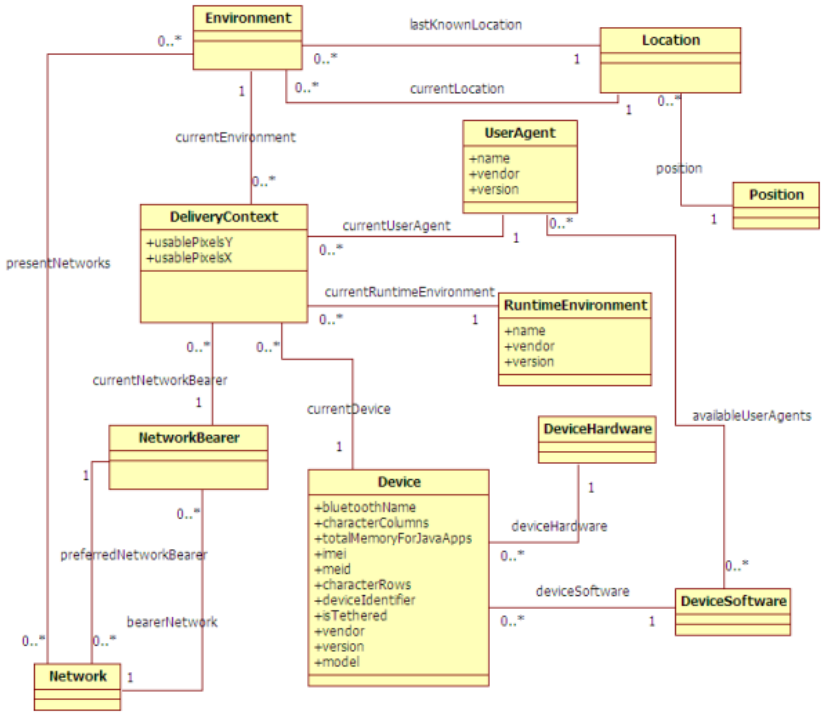


**Figure 6. Class d\Diagrams for Application Scenario for U-healthcare Applications**

**Scenario 2:** Modeling the Application Scenario for U-healthcare applications using class diagram as in Figure 6.

**Scenario 3:** CC delivers Spec produced through Service Agent to DC manager via SOAP. Also it receives DC Report including medical and resource Event corresponding to Spec, and delivers the Service Container (SC) Report to Event Handler one by one as in Figure7.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
- <ECReports ALEID="lit edge manger alpha" date="2005-09-01T14:36:33.359+09:00" specName="ECSpecName_define"
  terminationCondition="DURATION" totalMilliseconds="516" creationDate="2005-09-01T14:36:33.359+09:00" schemaVersion="1"
  xmlns="urn:epcglobal:ale:xsd:1">
- <reports xmlns="">
  - <report reportName="report1">
    - <group>
      - <groupList>
        - <member physicalLocation="127.654 37.551">
            <epc>urn:epc:tag:sgtin-64:4.011562.0557083.19206803</epc>
          </member>
        </groupList>
      </group>
    </report>
  </reports>
```

**Figure 7. DC Report Corresponding to the SC Spec**

**Scenario 3**: Event Handler of Event Layer receives the delivered Report, creates a logical event corresponding to the pre-determined Logical Event Rule, and delivers it to the corresponding Business Rule. Figure 8 shows a logical event (Current) representing the present status.

 Copyright ⓒ 2015 SERSC

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
- <LogicalEvents date="2005-09-01T14:36:33.359+09:00" creationDate="2005-09-01T14:36:33.359+09:00" schemaVersion="1"
    xmlns="urn:rclit:aaf:xsd:1">
  - <events xmlns="">
    - <event>
      - <current reportName="report1">
        - <group>
          - <groupList>
            - <member physicalLocation="127.654 37.551">
                <epc>urn:epc:tag:sgtin-64:4.011562.0557083.19206803</epc>
              </member>
            </groupList>
          </group>
        </current>
      </event>
    </events>
  </LogicalEvents>
```

**Figure 8. A Logical Event (Current) in DC**

**Scenario 4**: Business Layer executes a business process corresponding to the delivered Logical Event. And this layer represents the logical event corresponding to Business Process, and can transfer this event to MediaFeedItem or other legacy systems, too. In addition, this layer processes static information from MediaFeedItem which is given by manufacturing company, and track & trace information from ONS which contains information on the supply chain. Figure 9 shows a screen shot of a meaningful result of a logical event Service Container according to the business rule in application for mobile, and Figure 10 shows a meaningful result of a Service Container event according to the business rule in the CC-DC-SC accordingly.

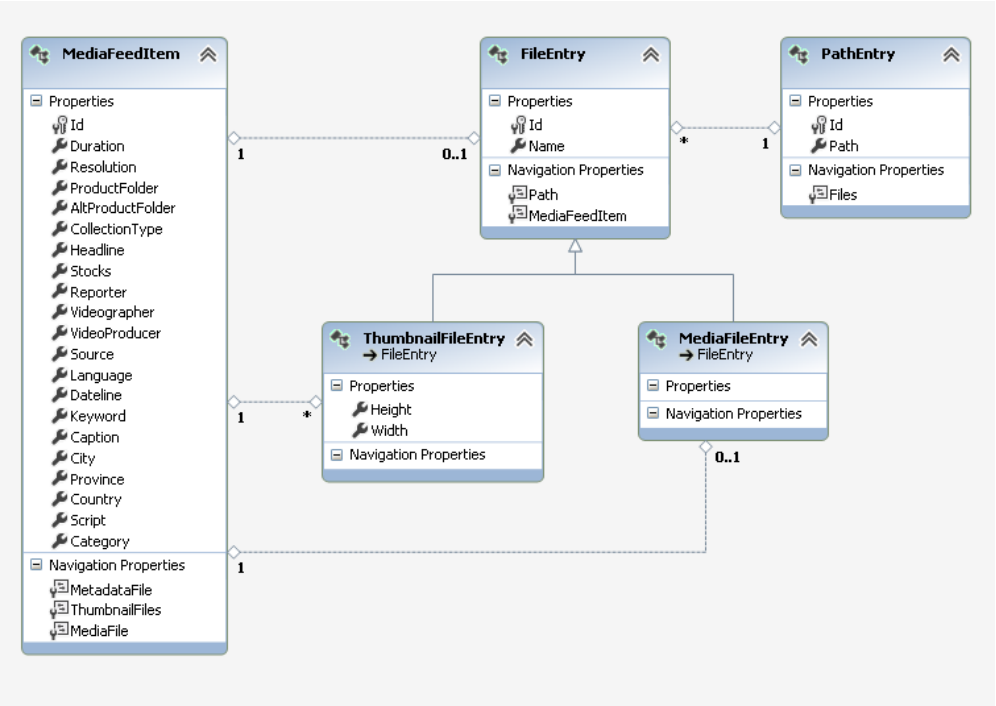


**Figure 10. A Meaningful Result of a Service Container Event**

Copyright ⓒ 2015 SERSC

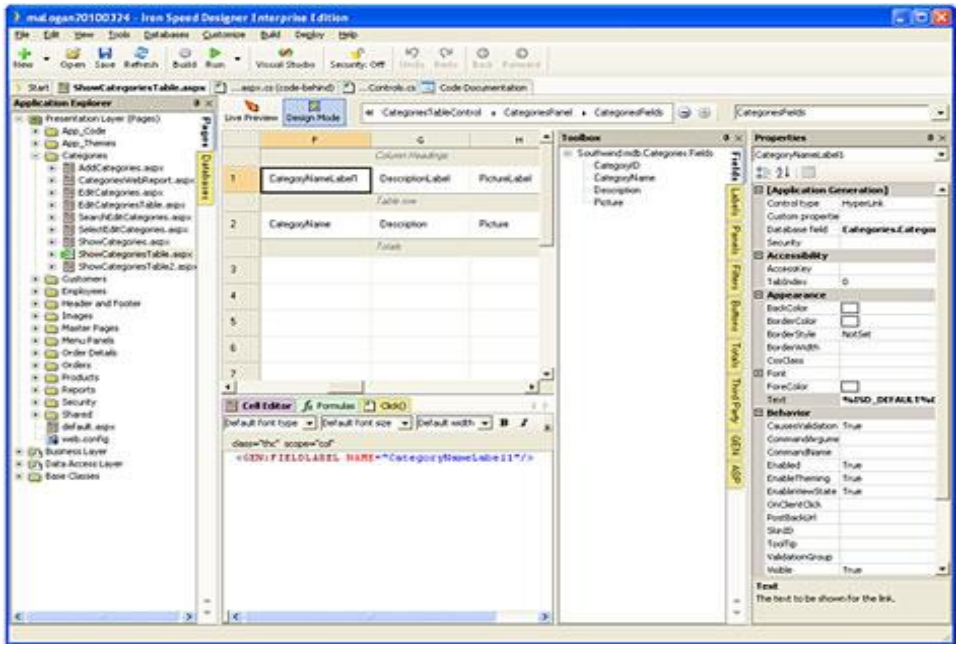

**Figure 11. The Storing Result of a Logical Event (Current)**

## 5. Conclusion

In this paper, we have proposed U-healthcare Enterprise Application Framework which is based on EPC global and RFID technologies, and have utilized CBD (Component-Based Development) and Mobile applications services to address the problem of the distributed computing over the Internet. Also we have demonstrated implementation of an example of RFID Application using AAF which is a light-version of EAF (Enterprise Application Framework). The characteristics of the Enterprise Application Framework are as follows: Application Layer may be modified when the domain is changed. Framework Layer, however, in the proposed system need not to be modified even if the domain is changed. Thus, we can reuse software which is fundamental technique in the software engineering.

We describe the requirements of u-healthcare Enterprise Application Framework, mobile applications development architectures and EPC global network which is RFID-based network environment. It include an architectures and scenarios how to construct the u-healthcare enterprise applications using the proposed Enterprise Application Framework by changing the contents of the Application Layer.

In the future, we will design and implement the detailed system for the proposed model in this paper. Also we will investigate the business application framework in more detail to leverage linkage between enterprise application and middleware or EPC related information.

## Acknowledgements

This research was Supported by the MSIP (Ministry of Science, ICT and Future Planning), Korea, under the CITRC (Convergence Information Technology Research Center) support program (NIPA-2014-H0401-14-1008) supervised by the NIPA (National IT Industry Promotion Agency). This research was also supported by the International Research & Development Pro-gram of the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT & Future Planning (Grant number: K 2013079410).

 Copyright ⓒ 2015 SERSC

## References

[1] M. De Jesus Mendes and F. M. de Assis Silva, "Mobile Agents in Autonomous Decentralized Systems", In Proceedings of the Fourth International Symposium on Integration of Heterogeneous Systems, Tokyo, Japan, (**1999**) March, pp. 258-260.
[2] S. M. T. Yau, H. V. Leong and A. Si, "Distributed agent environment: Application and performance", Inf. Sci., vol. 154, (**2003**), pp. 5-21.
[3] D. Heckmann, "Ubiquitous User Modelling", Akademische Verlagsgesellschaft Aka GmbH, Berlin, ISBN 3-89838-297-4.
[4] M. Harrison, "EPC Information Service-Data Model and Queries", *Auto-ID Center White Paper*, (**2003**).
[5] K. S. Leong, "EPC Network Architecture", *Auto-ID Lab*, (**2004**).
[6] M. Lee, B. Kang and D. Han, "Applications and Service (HEALTHCOM 2008)", 10th IEEE Intl. Conf. on e-Health Networking, (**2008**), pp. 244-249.
[7] F. Paternò, C. Santoro and L. D. Spano, "*MARIA*: A Universal Language for Service-Oriented Applications in Ubiquitous Environments", ACM Transactions on Computer-Human Interaction, vol. 16, no. 4, (**2009**) November, pp. 19:1-19:30.
[8] G. Meixner, N. Thiels and U. Klein, "Smart Transplantation - Allogeneic Stem Cell Transplantation as a Model for a Medical Expert System", Usability & HCI for Medicine and Health Care (USAB), LNCS 4799, (**2007**), pp. 306-317.
[9] EPCglobal, "The Application Level Events (ALE) Specification", Version 1.0 of (**2005**) February 8.
[10] C. Szyperski, "Component Software", *Addison-Wesley*, (**1998**).

Copyright ⓒ 2015 SERSC

Copyright ⓒ 2015 SERSC