

U-City Portal for Smart Ubiquitous Middleware

YONG WOO LEE, SEUNGWOO RHO

U-city Consortium, The University of Seoul, Seoul, Korea

The University of Seoul, Seoul, Korea

{ywlee, dairen}@uos.ac.kr

Abstract—Ubiquitous-city (u-city) middleware is essential to provide a variety of integrated and intelligent u-city services to u-city applications. Another essential element to u-city is the u-city portal if the u-city offers its services through World Wide Web. The u-city portal should well cooperate with the u-city middleware as well as support web-based applications efficiently. Yet, there is no reported research on a u-city portal based on a u-city middleware. We introduce u-city portal in this paper. This paper explains the mechanism, the interface to the u-city applications, the interface to the u-city middleware and how to cooperate with the u-city middleware to give a variety of u-city services to applications.

Keywords— U-city, U-city Middleware, U-city Portal

I. INTRODUCTION

U-city (ubiquitous city) refers to a city created through a marriage between the ICT (Information and communication technologies) industry, construction sectors, urban planning and management engineering. It has been enabled with a complex set of interlinked ICT-supported services.

It is usual that the u-city is controlled and managed through a central management center and the state-of-the-art communication infrastructure provides its residents real-time ubiquitous access to public services such as traffic information and weather news and commercial services such as distance learning in order to make citizens' life convenient [1].

Korea has been actively establishing u-cities which are spread across the Korean peninsula and our u-city consortium has been working for them. Seoul metropolitan government has been supporting our u-city consortium by funding five million U.S. dollars for five years and we have been doing R&D to develop related u-city technology and u-city products such as u-city middleware - it is called Smart Ubiquitous Middleware, SmartUM.[2] -, u-city portal, some applications and some infra-structures and apply them to u-cities.

The object of the u-city portal project is to provide an easy and consistent web interface to interact between the u-city middleware SmartUM in the processing unit and the u-city portal in the presentation unit. In other words, through u-city

portal, users can easily monitor and control all the resources and components that are managed by the u-city middleware in real time. Our u-city portal system can be flexibly combined with ubiquitous techniques for next generation ubiquitous computing because it is built based on enterprise techniques with scalability and independency. Through the u-city portal that plays an important gateway role of the u-city middleware, users are easily able to use the u-city environment even though they don't know complex mechanism of the u-city middleware and underlying infrastructure.

The outline of this paper is organized as follows: In section 2, we introduce Seoul Grid Portal that is our prototype portal for cloud computing and grid computing and now an important part of our u-city portal. Section 3 briefly introduces overview and architecture of the u-city portal in cooperating with our u-city middleware, SmartUM which consists of the four layers such as the Common Device Interface Layer, the Context-aware Computing Layer, the Ubiquitous Cloud Computing Layer and the Common Application Interface Layer. Section 4 explains the implementation of the u-city portal. Section 5 describes how to use the u-city portal in applications. Finally, section 6 gives conclusions and explains future works.

II. ENABLING TECHNOLOGY OF THE U-CITY PORTAL : SEOUL GRID PORTAL

We had built the Seoul Grid test-bed which covers whole area of Seoul, the capital city of Korea since 2002. We implemented a Grid resource management system into Seoul Grid Portal [3][4]. The Grid resource management system[5] contains real time performance monitoring subsystem, process management subsystem with resource brokering, account management subsystem, unified file transfer management subsystem and access Grid management subsystem. The Seoul Grid Portal had been built with GPDK [6] at the beginning and was rebuilt with GridSphere [7] later. Figure 1 depicts the Seoul Grid Portal using GridSphere. The u-city portal inherits many features of Seoul Grid Portal for Grid resource management.

A. Real Time Performance Monitoring (RTPM)

Seoul Grid Portal contains a real time performance monitoring facility. It monitors and shows two different Grid test-beds separately. They are RTPM-P2P (Real Time

Performance Monitoring-Peer to Peer) and RTPM-CC (Real Time Performance Monitoring-Cluster Computing).

B. Process Management with Resource Brokering

We see many job management systems currently. Some of them are PBS [8], LSF [9] Condor-G [10], LoadLeveler [11], and NQE [12]. However they do not fully support the Globus [13] based test-beds in job management. Thus we developed our own resource brokering system, namely, SGIRBS (Seoul Grid Intelligent Resource Brokering System) and implemented it into the Seoul Grid Portal. SGIRBS supports our Globus based Grid test-bed in seamless manner.

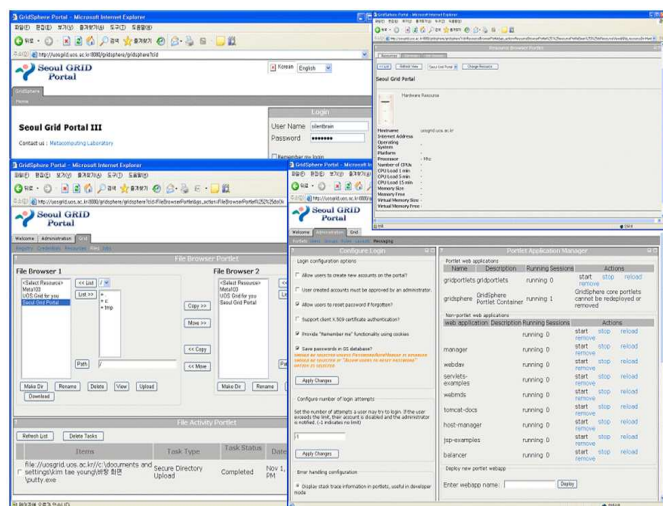


Figure 1. Seoul Grid Portal using GridSphere

C. Access Grid Management

The Seoul Grid Portal also supports access grid system [14] with the access grid tool kit. Pshell relays users' commands in users' PCs through remote consoles of AG nodes and the venue client. It also controls PC registry values through ActiveX and remotely controls the parameter values in the venue client

D. Unified File Transfer Management

Our file transfer management is a unified file transmission service that uses GSIFTP protocol and enables Seoul Grid Portal to transfer files to grid nodes. By using the GSIFTP, users could conveniently transmit required files to grid computing nodes.

E. Account Management

We developed our own user accounting system (SGIUAS: Seoul GRID Institute User Accounting System) using MS-SQL database, matched with the intelligent resource brokering function and implemented it into Seoul Grid portal.

III. ARCHITECTURE

Our approach to u-city services consists of 3 tiers, which are the presentation tier, the processing tier and the feeling tier as shown in figure 2.

A. The Smart Ubiquitous Middleware(SmartUM)

SmartUM in the processing tier is an intelligent ubiquitous-city middleware, which provides a data acquisition interface to sensors directly or through ubiquitous sensor networks and inference engine service that can process the collected data to make context-aware decision, that is, intelligent decision, therefore, optimal operations can be invoked and proper information can be generated.

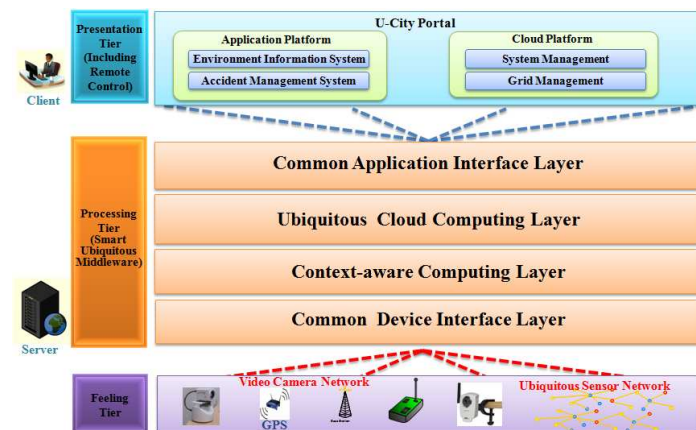


Figure 2. The three-tier architecture for u-city

The Common Device Interface Layer (CDIL) provides The Common Device Interface to devices at feeling tiers. The Context-aware Computing Layer (CCL) processes data obtained from the CDIL and provides the intelligent context to The Ubiquitous Cloud Computing Layer (UCCL). Also, it executes the reasoning process with its context. The UCCL performs intelligent services such as automatic service discovery, service deployment and service execution based on inferred contexts offered by the CCL in order to provide an automatic computing environment and enable applications or services to be used everywhere in a timely and cooperative way. The Common Application Interface Layer (CAIL) provides a set of applications such as the Environment Management, the Fire Accident Management and the Traffic Accident Management.

B. Ubiquitous City Portal(U-City Portal)

As described in figure 2, u-city portal that plays a gateway role is responsible for both presentation and remote control, and supports two main subsystems (platforms). One is the Application Management Subsystem which manages the environment information applications and the accident management applications. Those applications can be used as an independent system individually. The other is the System Management Subsystem which supports the system management and the Grid management. It manages various kinds of functions that are processed by SmartUM. The structure of the u-city portal is depicted in figure 3.

The Environment Information Applications shows noise information, air-pollution information and water quality information with geographical information. The Underground

Fire Accident Management Application uses SmartUM's functions such as the optimal path finder.

The Grid Management consists of many components such as Access Grid Manager, File Transfer Manager, Account Manager, Real Time Performance Monitoring Manager, and Resource Broker that have been used in the Seoul Grid Portal.

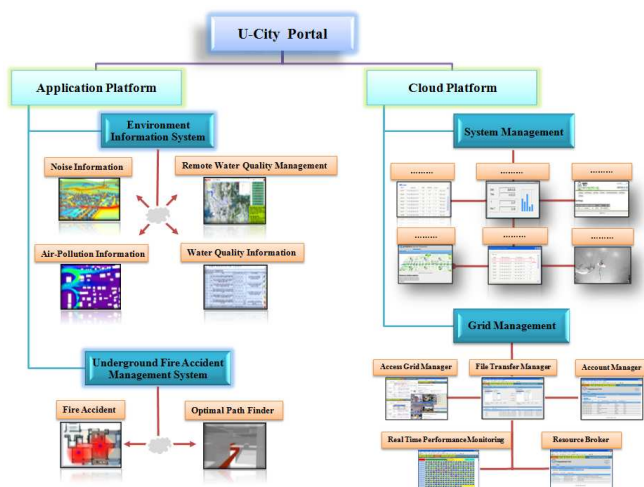


Figure 3. The structure of the u-city portal

IV. IMPLEMENTATION

We constructed the u-city portal with spring framework [15] since the spring framework provides many useful functions and light-weight solutions such as MVC, AOP, transaction management, RMI and so on.

All requests through the u-city portal have the form of URL. These URL requests call each controller with handler mapping information through the dispatcher servlet listener which looks like front gate. Each called controller only executes its logic that is pre-defined in SmartUM, and the result of the executed logic is transmitted to the JSP view. Finally, the requester looks the result in JSP view.

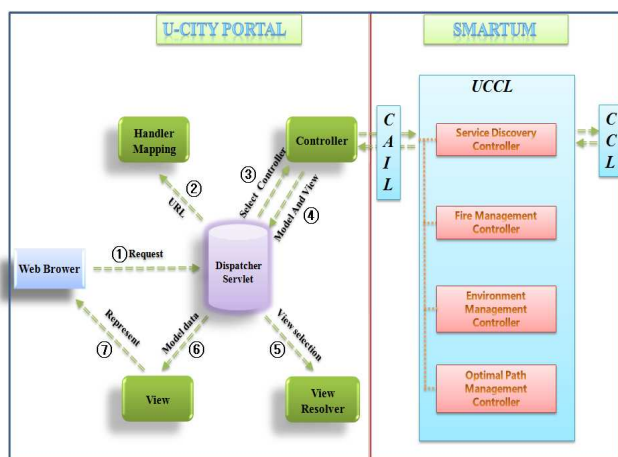


Figure 4. The workflow of Spring MVC through SmartUM

The figure 4 shows the workflow of the request process through the u-city middleware.

1) If a user requests URL to the u-city portal, the URL request is given to the Dispatcher Servlet. 2) The Dispatcher Servlet refers Handler Mappings to determine which controller would process the request. 3) It selects a controller which is suitable to the request. The controller processes one's logic in both the CCL and the UCCL of SmartUM. 4) As the result, the controller returns a model and a view to the Dispatcher Servlet. 5) The Dispatcher Servlet gets the view information from the ViewResolver. That is, after the Dispatcher Servlet receives the model and the view information, it requests the ViewResolver to look for real JSP. 6) The Dispatcher Servlet transmits the model data into the view object. Finally, 7) the response object displays the pages of the model data through the web browser.

When we use the u-city portal, we might need region information. In the case, users can see the region information in the form of either list or map. In order to show the region information on the map, we use open API in NaverMap [16] that is implemented with JavaScript. To use this open API, we must register the URL of the u-city portal in view code and take some key. If the URL is changed, former key is not valid. To solve this problem, we append some logic, that automatically changes the key, to the controller of the u-city portal. Also our portal can display the region information loaded from DB (Context Repository of SmartUM) on the map. We used the marker function which is provided with open API. If user clicks each marker, information of each region enlarges more and more.

The u-city portal is operated on the tomcat container of window server. Because we use web MVC of spring, we can separate control and view from data unlike pre-existing JSP programming cases. Figure 5 shows a snapshot of the u-city portal.

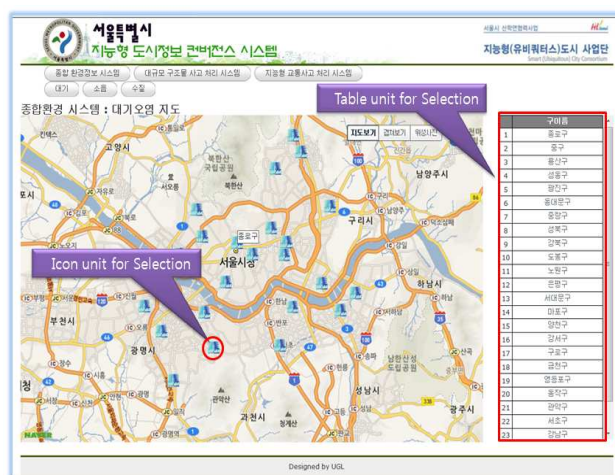


Figure 5. A snapshot of the u-city portal

We divided the u-city portal into M(Model), V(JSP), C (Controller). Because of such division, we can easily reuse each element. As well, we could quickly code our portal when we change some element. For example, we can simply change our portal by only replacing documents of properties such as

document related to SQL used in the DAO or to JDBC connection without recompilation. Figure 6 shows the sequence diagram of a request process in the u-city portal.

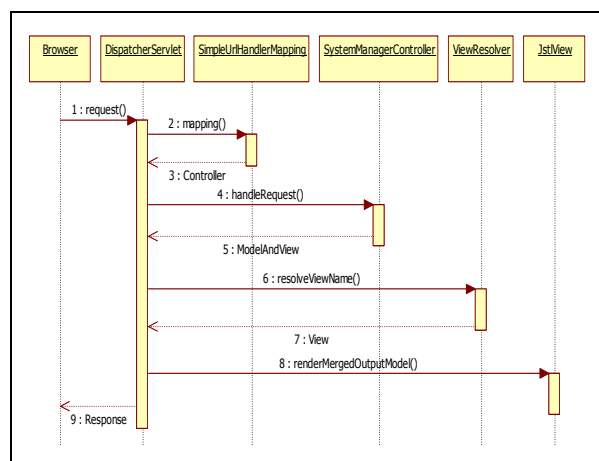


Figure 6. The sequence diagram of a request process

V. APPLICATIONS

A. Environment Management with 3D GIS Visualization

SmartUM provides intelligent services to various kinds of applications. One of the applications is the Environment Management Application using 3D geographical information. SmartUM is directly connected to the u-city portal to enable users to use the applications conveniently. First of all, the noise data is collected from the CDIL of SmartUM and saved in the Context Repository of the CCL. If user clicks any region on the map, the CCL calls the Service Broker in the UCCL.

After it, the Service Broker calls the Environment Monitoring Controller and forwards the data received from the CCL (Context Repository) and the region information requested from user, then it also forwards the data to the CAIL. Next, the CAIL executes the geographical visualization function with the noise data. Finally, the geographical visualization component creates a JPG and VRML file to visualize the data. As a result, the u-city portal shows it through view objects to the user.

B. Underground Fire Accident Management with Optimal Path Finder

The Underground Fire Accident Management Application can inform users the occurrence of fire and give optimal information through the u-city portal. In this case, the CCL converts the collected sensor values at the CDIL into ontology instances, makes inferred ontology instances with general ontology instances and saves event objects of the inferred ontology instances in the repository. While a user monitors each room in a specific building through the u-city portal, he may find fire event of a certain room. If the user clicks the room marked as 'fire status', the Service Broker calls the Optimal Path Management Controller that is one of the Service Controllers. Then the geographical visualization

component of the CAIL returns files of the optimal path finder to the u-city portal, and the u-city portal displays the visualization result to the user. Although the u-city portal processes these kinds of complex steps in the system, user doesn't need to know them because the u-city portal takes care of the process. If a user selects a specific floor in the building, he can see the sectional information and the floor information of the room. When fire is occurred, the color of the room is changed to be red. At the same time, button of the optimal escaping path is activated, thus user can easily find the optimal escaping path in case of fire circumstances by simply clicking the button.

VI. CONCLUSIONS

In this paper, we explained the mechanism and interface of the u-city portal, which is a part of intelligent u-city systems, the Utopia. Through the u-city portal, users could easily monitor and control all resources and components in cooperation with the u-city middleware, SmartUM in real time. Users do not have to be aware of the u-city, the underlying infrastructure and mechanism. Our portal model was implemented with spring framework and open API of NaverMap so that two platform management of the u-city portal can enable intelligent automatic processing to cope with the emergency and to manage some GIS visualization automatically according to predefined scenario in ubiquitous cities. The u-city portal project will be continued to support more heterogeneous applications and give more convenient and better u-city services. We will pay more effort to achieve the goal in future.

ACKNOWLEDGMENT

This study was supported by the Seoul Research and Business Development Program(10561), Smart(Ubiquitous) City Consortium and Seoul Grid Center. We would like to give thanks to Miss Hae Sun Jung, Mr. Chang-Ho Yun, Miss Ji-Sung Kim, Mr. Jong-Won Park, Mr. Hyun-Kyu Park, Mr. Jung-Dae Kim, Mr. Sun-Oong Lee, Mr. Cheol Sang Yoon, Mr. Chang-Won Lee, Mr. Phil-Doo Hong, Mr Kyoung-Gyu Lee and the staffs of Seoul Grid Center and the members of Smart (Ubiquitous) City Consortium for their contribution to this research.

REFERENCES

- [1] National Computerization Agency, "Toward a Korean U-City Model," *IT Strategies Issue 05-Strategy-09*, Sep. 2005.
- [2] H. S. Jung, C. S. Jeong, Y. W. Lee, and P. D. Hong, "An Intelligent Ubiquitous Middleware for U-City: SmartUM," *Journal of Information Science and Engineering*, vol. 25, pp. 375-388, Mar. 2009.
- [3] Y. W. Lee, "Seoul Grid Portal: A Grid Resource Management System for Seoul Grid Testbed," *Grid and Cooperative Computing - GCC 2004*, vol. 3251/2004, pp. 899-902, Sep. 2004.
- [4] Yong Woo Lee, Phil Doo Hong, "Web Service for Seoul Grid Testbed," *Computer and Information Technology*, 2006, pp. 63, Sep. 2006
- [5] Yong Woo Lee, Phil Doo Hong: A Job Management System for a Large Scale Grid System *Computer and Information Technology*, 2007, pp. 291-294, Oct. 2007
- [6] (2009) DOE Science Grid.: Grid Portal Development Kit website. [Online]. Available: <http://doesciencegrid.org/projects/GPDK/>
- [7] (2009) GridSphere website. [Online]. Available: <http://www.gridsphere.org/gridsphere/>

- [8] (2009) PBS website. [Online]. Available: <http://www.pbspro.com>
- [9] Ming Q. Xu.: Effective Metacomputing using LSF MultiCluster, *1st International Symposium on Cluster Computing and the Grid*, pp. 100-105, May 2001
- [10] J. Frey, T. Tannenbaum, I. Foster, M. Livny, S. Tuecke.: Condor-G: A Computation Management Agent for Multi-Institutional Grids, *Cluster Computing*, vol. 5, pp. 237-246, Jul. 2002
- [11] (2009) LoadLeveler website. [Online]. Available: <http://cantmc.canterbury.ac.nz/UCSC/LoadLeveler>
- [12] (2009) NQE website. [Online]. Available: <http://meta.cesnet.cz/cms/opencms/en/docs/software/system/nqe.html>
- [13] I. Foster, C. Kesselman.: Globus: A Metacomputing Infrastructure Toolkit, *International Journal of Supercomputer Applications*, vol. 11, pp. 115-128, 1997
- [14] (2009) Access Grid website. [Online]. Available: <http://www.accessgrid.org/>.
- [15] (2009) Spring Framework Reference website. [Online]. Available: <http://www.springsource.org/>
- [16] (2009) NaverMap Open API website. [Online]. Available: <http://dev.naver.com/openapi/apis/map/javascript/reference>