Next-Generation Library information service - 'Smart Library'

Byung-Won Min

Department of Information Communication Engineering, Mokwon University, Doan-dong 800, Seo-gu, Daejon, 302-729, Korea
minfam@mokwon.ac.kr

Abstract

For a library to be able provide information services and fulfill its function as a knowledge convergence center capable of responding to various information demands, the development of next-generation information systems based on the latest information and communication technology is needed. The development of mobile information services using portable devices such as smart phones and tablet PCs and information systems which incorporate the concepts of cloud computing, SaaS (Software as a Service), annotation and Library2.0 is also required. This paper describes a library information system that utilizes collective intelligence and cloud computing. The information system developed for this study adopts the SaaS-based cloud computing service concept to cope with the shift in the mobile service paradigm in libraries and the explosion of electronic data. The strengths of such a conceptual model include the sharing of resources, support of multi-tenants, and the configuration and support of metadata. The user services are provided in the form of software on-demand. To test the performance of the developed system, the efficiency analysis and TTA certification test were conducted. The results of performance tests, It is encouraging that, at least up to 100MB, the job time is approximately linear and with only a moderate overhead of less than one second. The system also passed the level-3 or higher criteria in the certification test, which includes the SaaS maturity, performance and application program functions.

Keywords: Collective Intelligence, Cloud Computing, Smart Library Information System, SaaS, Metadata

1. Introduction

As The change in the information environment caused by the advance of information and communication technologies has occasioned the need for new changes in the contents, systems and services of libraries and data centers. The information system in the Web 2.0 environment emphasizes the participation, sharing and collaboration of users through Social Network Services (SNS) such as Facebook, Twitter, Flicker and blogs [1, 2]. Furthermore, the convenient mobility of information services using smart phones and tablet PCs is creating a ubiquitous environment in which information can be utilized anywhere and anytime for decision making and problem solving [3, 4].

To perform the function of a knowledge convergence center and thereby respond to a wide range of information demands, the development of mobile information services using portable devices such as smart phones and tablet PCs and information system which incorporate the concepts of cloud computing, SaaS, annotation and Library2.0 is required [5-8]. Furthermore, the collection of data produced by the relevant agencies and digitalization of the collected data are needed to develop the contents services to cope with the changing information environment [9-11].
Many From a library perspective, as the advancement of computer and communication technology is making the Internet universally available, the ubiquitous environment, in which users can utilize contents anywhere and anytime while surfing on the sea of information, is realized. Users are using digital libraries with home computers or mobile devices over the wired and wireless networks. Such a ubiquitous environment also demands a new infrastructure to support the rapidly advancing terminals, networks and contents production technology [12].

Currently, most library systems use the client/server and ASP services for their software. However, such systems are difficult to manage and require high operating costs because of the problems related to high HW and SW purchasing costs, installation and distribution, customization, upgrade, fault and problem management, and expensive license royalties [13-14].

Furthermore, library systems are not widely accepted by users as they are unable to create the information and knowledge needed by them. Although library systems provide useful information such as announcements, newly-arrived book lists, Q&A, check-out status, recommended books, various research data, and new database data, such information is generally posted on the homepage as it is without filtering the information given by the information creators. As such, these systems are either difficult or limited for users.

Since the homepage provided to the users of a library system is centered on the library instead of the users, it is difficult even for users who are familiar with the library homepage to find the necessary information.

This paper proposes a technology for designing and deploying the smart library information system using collective intelligence and cloud computing to provide the needed information to the users by supplementing the problems described above.

This paper proposes a cloud-based integrated digital library system to overcome the problems of the existing digital library. It uses metadata to support customization for users. It differs from conventional methods in that it supports the tenants, which represent the user groups, with a software instance. It solves the weakness of ASP, which is not only costly to customize but also cannot take advantage of economy of scale because it loads each instance individually. Furthermore, there is very little initial investment since the library functions are standardized and modularized. They are designed and deployed as the cloud-based software on-demand type service model to enable easy, simple and low cost IT services.

2. Next-Generation Library Information Service Utilizing Collective Intelligence and Cloud Computing

2.1. Trend of Library Information Service Advancement

The As the growth of information is exploding without limits, understanding information and the meaning of information itself have become a more important issue than the amount of information. This breaks away from information in the system aspect as asserted by information theory, and reflects the new interpretation and utilization of information with the emphasis on the service aspect [11]. Not only library systems but also other similar information systems will have to provide information services correctly according to the frequently changing ‘context’ based not on the amount of information but on intelligent search, social search and context aware computing. To provide such an information service, it is expected that technology which incorporates semantic Web, pattern recognition and intelligent search to understand the information context and provide customized services will become increasingly important [12]. This section discusses the trend of information services
with the advancement of information technology, particularly collective intelligence and cloud services, which could become the basis of next-generation information service technology.

2.1.1 Collective Intelligence: Collective intelligence is based on the concept that “a group may achieve a task that cannot be accomplished by an individual.” Its significance is that it solves a problem using the intelligence of a group of people who communicate and collaborate with each other instead of relying on the ability of just a few people. By reducing the probability and cost of a failure that may occur when a certain individual performs a task or makes a decision, and by enabling the handling of more complex tasks by increasing the number of participants, it makes possible that which was not achievable by an individual. Its importance as a tool to provide a political and economic edge in the contemporary knowledge and creativity-centered global economic environment has increased significantly [13].

- Naver (NHN Corp.) is building collective intelligence into its blog service by providing a space for users to freely share and accumulate their interests and knowledge and by utilizing link-based clustering to allow other users to not only search but also to scrap and reprocess the contents. Furthermore, it provides a simple user interface by effectively applying trackback and RSS (Really Simple Syndication) to deploy the user participatory system naturally [14].

- Kyonggi University is developing a new collective intelligence-based context-sensitive information search technology. It enables an effective collective intelligence-based mobile information service under the ubiquitous environment by appropriately converging context-sensitive computing technology, Web 2.0 technology, and traditional information search technology. Moreover, it uses the automatic context tagging technique to deploy collective intelligence and effectively provides context-sensitive information according to priority with the folksonomy architecture [15].

- Konkuk University operates a wiki site in which librarians, researchers, professors, students and general users can participate in creating/revising and utilizing information. The site developed a type of source information sharing system combined with the wiki system to utilize collective intelligence [16].

- ‘Librarian Glossary 2.0 Created by Librarians’, a glossary collectively created by librarians, is the only library-related wiki service in Korea. Information about the library intended for librarians is being built in the form of collective intelligence in the era of Web 2.0 [17].

2.1.2 Cloud Service: The cloud service is a type of on-demand IT service in which users can receive IT resources over the Internet anywhere and anytime. Based on virtualization and distributed processing technology, it provides software, platform, infrastructure and other IT services to users by integrating various resources.

The cloud service has emerged as the answer to the demand for expansion of the IT service environment with the rapid spread of the Internet and the development of new Web services such as Web 2.0. The IT service environment is evolving from grid computing, which groups IT resources for application on the network, to utility computing and now to cloud computing. The cloud service is largely divided into IaaS (Infrastructure as a Service), PaaS (Platform as a Service), and SaaS: IaaS rents elements of the IT infrastructure such as the server, desktop
computer and storage needed for business operation in the cloud service form; PaaS receives the platform to develop the software needed for business operation through the cloud service; and SaaS, the subject of this study, rents the software in the cloud service form.

It is expected that the cloud service will be adopted by various organizations requiring computing resources and play the key role in not only reducing IT costs but also in improving IT resource efficiency and initiating new changes of business operation by overcoming various spatial and chronological limitations. For example, the cloud service can lower the computing operating cost, such as IT resource purchase and maintenance costs, and improve IT resource efficiency, as an organization can rent only the resources it needs. In other words, organizations which adopt the cloud service can quickly and flexibly respond to the rapidly changing business environment by utilizing the various computing services without the limitations of time and space. Furthermore, the cloud service can greatly improve business continuity through ‘smart work’ such as telecommuting and mobile office by extending the business operation space to the wired/wireless network space through the Internet. Eventually, it will have a significant impact on security and the reduction of social overheads such as environmental and energy costs [18].

2.2. Web 2.0 Based Library Information Service

Information search on a library homepage tends to be less effective than on other sites. To overcome this problem, libraries renewed their homepages, basing them on Web 2.0. As a result, library homepages became more user-centered thanks to the application of such technologies as AJAX, RSS, Open API, Mash-Up, Wiki, and blogs.

Although it is important to develop new tools or services for the delivery of information, what is more important is to create information specifically to be delivered to users. The latest information delivery tools make it easier to create information to guide users to utilize library information and stimulate the desire for such information at the same time. Eventually, it will present an opportunity for libraries to expand their role as information creators [19].

The majority of libraries including college libraries provide a wide range of information via their homepage. However, information services other than the search function have not gained a favorable response from users for various reasons, the most typical being that they can easily find the desired information over the Internet without recourse to a library. Indeed, most users identify the library as a space primarily for reading books.

To solve such problems, libraries are providing digital reference services such as real-time chatting and e-mail services. Although a homepage provides the information service in various ways, limitations arise because of its inherent problems: it forces users to visit the library homepage, thus requiring them to know which information they need, and provides only the library’s data. In the meantime, portal sites and other websites have adopted cutting-edge Internet technologies to improve their services by making them easier to use, thus eliciting a favorable response from users.

Libraries are also attempting to apply such technologies to improve library systems and services. The introduction of RSS in particular helped to make library contents available outside and hence accessible from anywhere. Consequently, users can reduce the time spent searching for the library homepage. Moreover, a blog can improve communication between librarians and users, and should be considered as a tool for enriching the user’s experience. Librarians who do not actively utilize it will be left behind in the era of the digitalized library.
However, studies on library services focus on RSS and blogs, which are the leading Web 2.0 technologies, while there are few studies on library services using other technologies. New studies proposing a possible operation direction for librarians at college libraries and studies on how some librarians utilize RSS and blogs for the library information service are just beginning.

2.3. Next-Generation Library information service - 'Smart Library'

The library information service of the future must go beyond just providing passive and simple knowledge accumulation and become a smart library that facilitates active knowledge creation and provides services through collective intelligence. Figure 1 shows the basic concept of a smart library.

![Figure 1. Overview of Smart Library](image)

First, Most libraries nowadays create a wide range of information and provide it to users in addition to managing the information. Although libraries provide user training, announcements, etc. through their homepages, the information is often not delivered to users. However, some libraries recently adopted Web 2.0 to provide information, and that helped to properly deliver information to users. Therefore, which information to create and provide to the users has become a key issue for libraries. If libraries only continue providing information on new arrivals, announcements and databases, they may not gain the attention of users.

Therefore, the future library information service must be a smart library which combines the existing library system functions with the knowledge-based e-learning system to develop creative human resources. It must be a system that enables the participants to develop their collective intelligence-based learning knowledge base using a collaborative and interactive interface, and which promotes self-managed learning to improve the creativity and logical thinking of the student.
While the existing library information service emphasizes the 1:n online knowledge service provided by library personnel, the smart library system can create, validate and categorize knowledge using collective intelligence group and provide intelligent knowledge, realistic knowledge, customized knowledge, and hands-on knowledge.

Furthermore, it enables contents sharing and interchanges of opinions using collective intelligence. The learning contents and knowledge base developed by collective intelligence can improve the national knowledge resource competitiveness, and smart tutoring in the next-generation e-learning environment can help achieve the national policy goals of developing creative human resources, improving public education, reducing private educational expenses, achieving a more balanced distribution of learning opportunities, and overcoming the gap between regions and social classes.

The smart library can play a central role in improving the country's position as a knowledge powerhouse in the era of the knowledge-based society by realizing nationwide lifelong learning and raising the educational level of the general public. The application of ubiquitous technology will enable the building of student-centered learning spaces and the restructuring of classrooms, and promote collaboration and cooperation between government agencies on the construction of a ubiquitous-based lifelong learning system.

2.4. Smart Library Information Service System

The main reason why users do not actively respond to the information provided by libraries is that the latter have failed to create the information and knowledge needed by them. Although the library system provides useful information such as announcements, newly-arrived book lists, Q&A, check-out status, recommended books, various research data, and new database data, such information is generally posted on the homepage as is without filtering the information given by the information creators. As such, it is difficult to use or somewhat limited as far as users are concerned.

The smart library system of the future must allow the collective intelligence group to reprocess information so that users can easily understand and solve such problems. New tools designed for the efficient delivery of library knowledge and information must also be developed. Figure 2 below is a schematic diagram of a smart library service.

The library information service of the future must be developed with a new system so as to feature useful information/knowledge and to deliver it efficiently. If libraries do not create information and knowledge that promote library use, there will be no change in the user's perception of the library. Libraries must now extend the role of the service at the information management level to that of a knowledge base convergence repository to create and manage information and knowledge. Information and knowledge must go beyond being only of a guidance nature to stimulate users’ desire to learn and to promote interest in libraries’ data.

As shown in Figure 2, a smart library information service system provides the service converged with the intelligent tutoring system, as well as adding an intelligent learning engine, collective intelligence interactive interface, knowledge-base manager, and open framework technology to the conventional library information service system.
An ‘intelligent learning engine’ is an engine that improves students’ creativity and logical thinking by providing them with the optimum learning process and knowledge contents intelligently, supporting the repetitive/dynamic learning process, and diagnosing their cognitive affective nature, thereby enabling them to engage in self-managed, explorative learning.

A ‘knowledge base manager’ is a management tool designed to systematically manage collective intelligence knowledge and to accurately and quickly search such knowledge in order to provide a self-managed and explorative learning environment.

A ‘collective intelligence interactive interface’ is an interface that supports experts’ efforts to create knowledge content, provides various visualizations of the created knowledge, and supports interactive interaction between the users and the system.

An ‘open e-learning interoperable framework’ is a framework that provides a platform for e-learning interoperability and integrates legacy services.

An ‘intelligent collective intelligence interactive learning agent’ promotes learning and teaching to improve creativity and logical thinking by providing an interactive interface that promotes cooperation among the participants, a tutoring operation model for collective intelligence and cooperative learning, and augmented reality, virtual reality and explorative learning pattern analysis.

The ‘learning platform and teaching/learning strategy model that supports the intelligent knowledge and composite/dynamic-adaptive learning model’ provides a learning platform through which students can receive the optimum education by dynamically restructuring and delivering the learning model - such as the learning contents, progress and instructor -
according to the student’s level, achievements, and inclination, as well as the level of difficulty of the contents.

Therefore, a smart library system is a next-generation, integrated learning system designed to cultivate and develop creative human resources. It is an intelligent tutoring system that uses a cooperative interactive interface to build a collective intelligence-based learning knowledge base and induce/support students’ self-managed learning in order to improve their creativity and logical thinking.

3. Development of the Smart Library Information Service System

3.1. LinkSaaS Application Platform

LinkSaaS Application Platform is software infrastructure designed to tame the data deluge with cost-effective, supercomputer-like capacity to process, transform, and indeed channel it for our benefit. It does this by leveraging the spare capacity of computing resources on the “edge” of the Internet – resources which generally use only a small fraction of their capacity even when busy and are available for marginal operating cost and environmental impact – and allowing users and developers to combine and coordinate them in novel, dynamic ways. In order to assure the safety of these edge resources, LinkSaaS Application Platform builds upon Java applets, the best established security context for running untrusted, third-party code on the Internet; the use of this technology also seamlessly extends the reach of LinkSaaS Application Platform across different platforms, operating systems, and even enterprises. In principle, the LinkSaaS Application Platform network could encompass all the world’s network-attached computers, providing an aggregate capacity 100 times greater than the 500 most powerful supercomputers combined.

The purpose of the LinkSaaS architecture is twofold: first, LinkSaaS is designed to increase the productivity of the installed base of computing devices. As noted above, most of the computers in the world use very little of their computing capacity even when “busy”, e.g., editing a letter or spreadsheet, or browsing the Web. LinkSaaS aims to harness this spare capacity for general computing. However, there are significant technical obstacles to realizing this purpose. For this reason, LinkSaaS is designed to be simple to join, leave, and administer, with the resource owners remaining firmly in control of their computers; moreover, it must satisfy reasonable security expectations for running untrusted, third-party code.

Second, LinkSaaS is intended to increase the productivity of developers creating new applications for this large network. Distributed programming historically has been problematic and fraught with tedious details – in short, not usually worth the effort except in extreme cases. The easiest situation is where the developer only has a single computer cluster under his influence, but this cluster may not be large enough. Once administrative boundaries are crossed, even within the same organization, the developer must now keep track of a variety of configurations, accounts, and security mechanisms. In addition, the code usually has to be built and installed on platforms which all differ, even if only slightly (hence the tedious details). The code then has to be activated, and the network configuration itself, such as the presence of firewalls, must be considered.

LinkSaaS addresses the above problem by specifying a streamlined environment for programming and deploying code across the network based on Java applets. This
environment will be described in greater detail below. With LinkSaaS, developing, deploying, and provisioning a large-scale network application becomes realistic and practical.

3.1.1. Description: In this section, the LinkSaaS architecture is described. Its components are described first, followed by descriptions of how developers, users, and administrators see the resulting system. Every attempt has been made to keep LinkSaaS simple and straightforward to join, leave, program, and administer.

The LinkSaaS network itself provides a basic infrastructure which makes the network programmable much like a peer-to-peer network while still respecting administrative constraints. It is not intended to make a distributed system look like a single computer; the authors suggest that this level of abstraction is best approached through software built on the LinkSaaS infrastructure. Moreover, the single computer abstraction is neither necessary nor appropriate in many circumstances; instead, programmers should be free to use the network creatively, and to construct network configurations appropriate to the nature and scale of the problems they are tackling.

To the extent that the LinkSaaS architecture is based on services which can be combined to form other services, it will be seen to bear some relation to a service-oriented architecture (SOA). However, LinkSaaS services do not have to reside on big, expensive servers; instead, they are provisioned by resources on the edge of the network. Also, unlike a typical SOA, LinkSaaS specifies a streamlined remote service deployment model, facilitating dynamic, highly distributed service provision; it thus relieves the programmer of some of the most tedious aspects of practical distributed programming.

Furthermore, LinkSaaS relies on the Java Virtual Machine (JVM), with its built-in security mechanisms, to accomplish remote deployments safely; it should be noted that Java applets have been running untrusted, third-party code in web browsers for over ten years. The JVM allows compiled code to run under Windows, Linux, Mac OS, and a number of other, less familiar operating systems, as well as on hardware ranging from the typical x86-based PC to an ARM-based mobile phone. The speed penalty for running code in the JVM has also largely vanished due to HotSpot runtime compilation into native machine code. By building on Java technology, LinkSaaS can offer secure, high-performance, cross-platform computing.

3.1.2. Components: Figure 3 shows a block diagram of a LinkSaaS network. The LinkSaaS network is divided into administrative domains, each of which can have one or more LinkSaaS servers. Each server is in turn connected to the Internet, through which it communicates with other servers as well as generic web servers. LinkSaaS clients then connect to one or more LinkSaaS servers; in principle, they could connect to servers in different domains, if their network access allows it.

It can be seen that the LinkSaaS network is an overlay network built, firstly, on the Internet itself. LinkSaaS servers are identified publicly by their Internet addresses, and since LinkSaaS servers and domains are fundamentally independent of one another, the LinkSaaS network is scalable in the same sense that the Internet itself is scalable. The network within a LinkSaaS domain may be different, however, and the clients may or may not have direct access to the
Internet – for instance, because of a firewall, or its implementation on a proprietary network. Even so, the LinkSaaS clients will remain accessible to LinkSaaS network operations.

Figure 3. Block Diagram of the LinkSaaS network, showing servers, clients, and application/web servers

3.1.3. LinkSaaS Server: The LinkSaaS server is the lynchpin of the LinkSaaS network: clients join the LinkSaaS network by connecting to a server, and the server acts as a proxy for the clients in communication with the wider network as well as with other clients (Figure 4).

In this way, the clients can communicate with the wider network even if they are protected behind a firewall – only the server need be exposed to the wider world. Within an administrative domain, a LinkSaaS server should be maintained and monitored as any central service, such as an e-mail service or an institutional web server. In turn, because of its central role in the LinkSaaS network, the server can act as an administrative control point for its clients, monitoring LinkSaaS bandwidth usage and limiting it if necessary.

Thus, even though clients can communicate with one another as if they were participating in a peer-to-peer network, the administrator can maintain a useful degree of control over their resource usage. The LinkSaaS architecture therefore addresses the common shortcomings in peer-to-peer network operation.

The LinkSaaS server manages the namespace of its domain, or, in other words, it assigns names to the clients connected to it. These names are globally unique by construction as long as LinkSaaS servers maintain unique IP addresses – which is an assumption of the Internet itself.
3.1.4. LinkSaaS Client: LinkSaaS clients are the workhorses of the LinkSaaS network: they are the programmable elements out of which developers create applications. The client itself is a program which runs on a host computer: it can be started from a web browser (applet client), run as a user program (stand-alone client), or installed as a persistent service (enterprise client). It is possible to have several clients running on a single computer; each client will have a unique identifier with which it identifies itself to the LinkSaaS server.

As noted above, a LinkSaaS client downloads its first service descriptor from a LinkSaaS server. This service descriptor tells the LinkSaaS client what to install in its root container. In the case of a stand-alone or enterprise client, the client can connect to more than one server simultaneously, initializing a root container with a different service descriptor for each – since each LinkSaaS server, potentially in different administrative domains, may implement different domain policies. It is the responsibility of the LinkSaaS server to decide whether or not to accept a connection from a particular LinkSaaS client.

The security context of an applet client is shown in Figure 5. In this case, the entire LinkSaaS client is encapsulated within a Java applet and its Security Manager.
3.1.5. **LinkSaaS Application Server**: LinkSaaS clients are able to access normal web resources via the LinkSaaS server. This feature allows them to access not only the whole World Wide Web for information or other functionality, but it also allows the developer a convenient method to control his application.

As noted above, LinkSaaS does not specify a particular application model, but a particularly simple model, used already for several applications, consists of a web server (the application server) which provides a work queue and a user interface through which a user submits to it. Clients are set up to poll the application server with simple HTTP GET requests for work to do. Results can be sent back via HTTP POST operations or stored until fetched and cleared by the application server.

The application server is not a part of the LinkSaaS network per se, and indeed can be written without any reference to LinkSaaS code; its interaction with LinkSaaS clients can be managed entirely through normal web and socket connections made through the LinkSaaS server. The server therefore could be implemented using simple CGI scripts on an existing web server, though more flexibility may be obtained by using streamlined web server infrastructure software such as Grizzly or LinkSaaS’s own NIONetworkHandler class. It should also be noted that a LinkSaaS service could also function as the application server.

3.1.6. **LinkSaaS Service**: A LinkSaaS service is a program that can be installed and run on a LinkSaaS client. Services can perform data processing, serve web content, and communicate with other services or external web sites. In addition, a LinkSaaS service can create child containers into which other LinkSaaS services can be installed.

It is worth noting the terminology: a LinkSaaS service runs on a LinkSaaS client, not a server. However, it can be said that a LinkSaaS server (which after all provides the root of the
service’s public URL) is what offers the service – even though the actual execution of the service is on another computer.

A LinkSaaS service is initialized by pointing the client to a web-accessible service descriptor in the form of an NML document. This descriptor can be located on any web server, not necessarily the application server or any LinkSaaS server (though the NML document describing the root container resides on the LinkSaaS server by design).

The LinkSaaS Markup Language (NML) has an XML-like syntax, with three tags defined:

- `<LinkSaaS-app>` defines the application’s name and encompasses all the classloaders and services it uses.
- `<classloader>` defines the location of class hierarchies and JAR files for services defined within its scope.
- `<service>` defines an actual service with a name and a main class. Data between the opening and closing tags are passed to the service’s initialization method as a character sequence (CharSequence).

The `<classloader>` tag is optional; any services defined outside a `<classloader>` scope is assumed to find classes starting from the service descriptor’s URL directory. It is often convenient for any necessary class or JAR files to reside alongside the service descriptor, but the `<classloader>` tag allows applications to draw upon classes from a variety of sources.

An example of an NML document describing a CommandService, an often-used root container, is shown below.

```
<--! Last touched 18-03-08 -->
<nerus-app default="command">
  <service id="command" code="org.nerus.commandservice.CommandService">
    a7a33b8eb14a49faa6e4af0b0b0f07
  </service>
</nerus-app>
```

**Figure 6. NML Document for CommandService Descriptor**

In this example, the application is called “command”, and its Java classfile is found relative to the server’s default web directory. The data between the `<service>` tags (in this case a primitive authorization string) is passed to the service on initialization.

An NML document can describe multiple services with different code sources. Services may be combined into larger services by including references in the initialization data, whether to other service names within the same NML document, or using URL’s of external services. An “auto-configuration” LinkSaaS service may be employed to generate further service descriptions as needed.

### 3.1.7. LinkSaaS Api:

The API of a LinkSaaS service is much like that of a Java applet: if it is deployed in a standard JRE, then it can use any of that JRE’s libraries. However, the usual applet security restrictions apply: it cannot modify system properties, cannot access the local filesystem directly, and cannot open network connections except to the server from which the
applet was originally downloaded. Actions beyond these must be authorized by the client (and by implication, the user) through Java’s extension mechanism.

The most basic LinkSaaS service interface (LinkSaaSService) specifies only an init() entrypoint, which is called when the service is created. The name of the service is determined by the service descriptor; the client delegates URL’s with this path element to the service.

An AbstractService class has been provided to make it easier to write new LinkSaaS services. AbstractService provides two abstract entrypoints, prepareToListen() and handleConnection(). prepareToListen() is invoked when the service is created (it is invoked by the AbstractService.init() method). handleConnection(), on the other hand, is called whenever a connection is made to the service from the outside, including the times when a user clicks into the service through the LinkSaaS server.

Because of the LinkSaaS client’s applet-like constraints, several methods have been provided in AbstractService to facilitate communication with the outside world. First, println() and printErr() methods have been provided to print messages to the output and error panels in the LinkSaaS client’s user interface. Web access must be redirected through the LinkSaaS server, which serves as a web proxy for its clients: a createRedirectedURI() method is provided to create the proxy URL from an original URL. A socket proxy is also provided.

The common application pattern described above is implemented on the service side by using prepareToListen() to create a new thread which periodically polls the application server (whose web address has been transformed by createRedirectedURI()) for work to do. In this case, handleConnection() can be used by the application server to fetch stored results, or by other LinkSaaS services to fetch intermediate results. It can also respond to requests for status information, for instance by users clicking through the LinkSaaS server web page.

3.1.8. **LinkSaaS Network and Namespace:** One of the key benefits of the LinkSaaS network to network programming is to define a straightforward namespace by which elements of the network can be addressed. The LinkSaaS namespace consists of URL’s, all of which are valid for public web access.

Each LinkSaaS server has a public URL such as http://domain1/, where “domain1” refers to the LinkSaaS server host. Since a LinkSaaS domain will typically have one or a few LinkSaaS servers, and policies are determined on a domain level and implemented by the servers, it is appropriate (though not necessary) to designate LinkSaaS servers with domain names. Since the LinkSaaS servers are publicly accessible Internet nodes, their names must be registered in the normal way.

The LinkSaaS server provides several web pages, such as http://domain1/clients, which lists the clients connected to it.

When clients connect to a LinkSaaS server, it creates a root container into which the default service, described by DomainRoot.nml, is installed. This service is addressed by appending the client alias to the clients subdirectory of the server URL, e.g., http://domain1/clients/client5. The client then delegates the interpretation of any further path elements to the service. Thus, if the service has created child containers in which it has installed services, those services are referred to in further path elements.

In principle, the entire LinkSaaS namespace is accessible via the Web. If a service needs to address another service, it creates a redirected URL out of the other service’s public URL, just as if it was a normal web resource.

3.1.9. **LinkSaaS Application:** Software delivered as a service offers distinct advantages over software delivered by more traditional means: it is frequently mobile, web-based, centrally managed, and nearly free of troublesome installations and patches. In its simplest form, the
architecture implied by these features consists of many clients attached to a large server infrastructure. Increasing numbers of users as well as improved functionality generally impact the server load and therefore drive server requirements, with increasingly expensive outcomes. Reducing server load therefore decreases the overall cost of the system, but risks substantially increasing operational costs.

![Image: Interactions within a LinkSaaS Application](image_url)

**Figure 7. Interactions within a LinkSaaS Application**

### 3.2. Development Scope

The system proposed in this paper consists of a total of forty-four modules, including twelve on the cloud-based library automation system, four on electronic source management, three on the search engine, three on language resources and others, sixteen on the collective intelligence-based smart learning system, and six on the interface.

The cloud-based library automation system is a module that performs the main functions of the integrated digital library system; it includes received book management, catalog management, check-out/return management, periodicals system, article index system, integrated search system, original document copying service system, personalized information service, research report management system, and contents management system.

The electronic source management includes NOS (NDSL on Site), BOS (Books on Site), ERM (E-Resource Management), and IRS (Institutional Repository solution for dSpace), etc.

The intelligent learning engine consists of learning self-diagnosis technology, knowledge contents recommendation technology, learning process recommendation technology,
knowledge presentation technology, knowledge systematization/organization technology, knowledge search technology, and knowledge DB management technology.

- The student diagnosis technology consists of student modeling technology, learning result analysis technology, student capability diagnosis technology, and cognitive affective attribute technology, all of which are used to assess the student using the smart learning system.

- The knowledge contents recommendation technology recommends the knowledge contents customized to the student-based student assessment and includes intelligent knowledge creation technology, optimum knowledge contents inference technology, and composite knowledge contents packaging technology.

- The learning process recommendation technology consists of formal learning process management technology, informal learning process creation technology and optimum learning process inference technology. It recommends the learning program to the student.

- The interface to collective intelligence technology includes the collective intelligence interactive interface, interactive guide agent technology, and interactive interaction technology.

- The interface to the legacy library system includes the collective intelligence interactive interface, knowledge creation tool, and knowledge visualization technology.

3.2. Development Environment

A smart library information service system development environment is composed of the following three layers: the first layer provides the LinkSaaS API in the SaaS platform; the second consists of Spring Framework, AOP, ORM, DAO, MVC, Web, Context, etc.; and the third layer, which is used for application development, consists of Web (JSP, JSTL, HTML, CSS, JavaScript, and XML), Business Logic (book receiving, catalog, search, article index, continuous periodicals, etc.), iBatis, and Oracle [20-22].

Spring Framework is an open source framework created to support the development of complex enterprise applications. One of the strengths of Spring Framework is that it is a layered architecture. Its main advantages are that a consistent framework is provided for J2EE application development and that users can selectively use only those components they need at the same time.

iBatis enables object orientation by separating the business logic and database access. It ensures flexibility of data processing, as the SQL statement and code can be independently generated. It also provides the transaction and cache functions.

The development environment for a smart library information service system that supports the multi-tenant environment is shown in [Table 1]. It uses Linux5.0 as the operating system, Intel Xeon Dual Core Processor 2.0GHz CPU, 2GB ECC DDR SDRAM memory, and a 146GB/10k rpm SAS Disk * 4 internal disk.
Table 1. Development Environment

<table>
<thead>
<tr>
<th>Item</th>
<th>Detailed Specifications</th>
</tr>
</thead>
<tbody>
<tr>
<td>Process</td>
<td>· OS: Linux 5.0 or higher</td>
</tr>
<tr>
<td></td>
<td>· Type: Intel Xeon Dual Core Processor</td>
</tr>
<tr>
<td></td>
<td>· Clock Speed: 2.0GHz or higher</td>
</tr>
<tr>
<td></td>
<td>· Quantity: 2 or more (up to 2 expandable)</td>
</tr>
<tr>
<td>Memory</td>
<td>· 2GB ECC DDR SDRAM (8GB Memory or more) or more</td>
</tr>
<tr>
<td>Internal Disk</td>
<td>· Capacity: 146GB/10k rpm SAS Disk * 4 or higher</td>
</tr>
<tr>
<td></td>
<td>· Disk Bay: 6 or more</td>
</tr>
<tr>
<td>IO Slot</td>
<td>· 6 Hot-Swap PCI-X 64bit slots or more</td>
</tr>
<tr>
<td>LAN</td>
<td>· Dual 10/100/1000Mbps Ethernet Controller(TP RJ-45)</td>
</tr>
<tr>
<td>Graphic-Card</td>
<td>· VIDEO: 8MB SDRAM or higher</td>
</tr>
</tbody>
</table>

3.3. Consideration

To provide a user service that emphasizes Web service search, accessibility and usability, contents creation and recombination, interoperability and a personalized service, the smart library must be developed within the concept of Web 2.0 and Library 2.0.

To develop a smart library information service system, an open Web conforming to the W3C standard is needed first of all.

- Web page supporting all browser versions of Netscape 7.x, MS IE 6.x, and Firefox 1.x or higher (cross browsing supported)
- Various operating environments: Web accessibility and usability
- Web service environment with consideration to generality and scalability
- Standardization: Mutual interfaces of the Web service and integrated usage
- “Administrative Agency Homepage Development/Operation Standard Guidelines”
- Web service standard architecture: XML, UDDI, WSDL, and REST
- “Information System Development/Operation Technical Guidelines 2.0”

Secondly, various mash-up services using open API (Application Programming Interface) must be considered.

Lastly, user authentication and access privilege management must be provided using the homepage login linking (SSO).
4. Analysis of the Performance of the Proposed System

4.1. Efficiency Analysis

To measure the performance of the smart library information service system supporting the multi-tenant environment, a simulation environment consisting of ① Web server, ② DBMS server, ③ media server, ④ exchange server, ⑤ IPPBX server, ⑥ client 1, and ⑦ client 2 was constructed.

IIS 6.0 was used as the ① Web server in which the proposed server module is installed. The proposed server module and MSSQL 2005 were used for the ② DBMS server. The proposed server module and Adobe Media Server 3.1 were used for the ③ media server. The proposed server module and the Red5 Media Server 0.7.0 were installed in the ④ exchange server. The proposed server module and Asterisk 1.4.21 PBX were installed in the ⑤ IPPBX server. The proposed client module, Internet Explorer 6.0, MS-Office 2007, Haansoft 2007, ViRobot Desktop v5.5 were installed in ⑥ client 1 while the proposed client module and Firefox 3.0 were installed in ⑦ client 2 for the client module test.

To measure the system’s performance, TeamQuest 10.1 Manager was installed in the ① server and TeamQuest 10.1 View in ⑥ client 1. The programs measured the server resource usage. LoadRunner 8.1 was installed in ⑥ client 1 to test the load generation.

To analyze the system’s efficiency, a scenario consisting of ‘measuring the server resource utilization rate, server memory usage and server response time after logging in an digital library system designed for 100 concurrent users and performing the registration, modification, removal, and search of the data’ was devised. The resource utilization rate, represented in %-processor time, indicates the percentage of time taken to run a non-idle thread. Memory usage, represented in ‘private Mbytes’, is the amount of the memory allocated to the process run by the computer. The server response time is measured in terms of the number of seconds it took to complete a command from the time a command such as system search or order command was input.

The figure below shows the result obtained for the server resource utilization rate (%) and the average memory usage of the server under the analysis scenario. Although the maximum resource utilization rate of the server could temporarily increase to up to 7.81% under the scenario-1 condition, it returned to stable conditions after the function was executed. The average server memory usage was measured to be around 803MB. The CPU and memory resources of the proposed system showed very stable utilizations.
when running the analysis scenario, and data management was also very stable under the conditions.

The results of performance tests, from searching text without the benefit of an index, are shown in Figure 8. It is encouraging that, at least up to 100MB, the job time is approximately linear and with only a moderate overhead of less than one second.

![Total job time vs Amount of input data (Smart Library Application)](image)

**Figure 8. Job performance of Smart Library application**

4.2. TTA Certification Test

4.2.1. Test Items: As shown in Table 2, the various items for the TTA certification test of the proposed smart library information service system comprise four items in SaaS maturity, three items in performance, and thirteen items in application program function.

The SaaS maturity test items include multi-tenant support, same code support, customizing support, and data sharing support. The performance test items include the CPU utilization rate, memory utilization rate, and response time. The application program performance test items include the received book system, catalog system, inspection system, academic journal table of contents system, periodicals, original document copying service, system management, DL contents management, research report management, contents management, original copy management service, ETM service, and general user service.
Table 2. TTA Certification Test Items

<table>
<thead>
<tr>
<th>Item</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>SaaS Maturity</td>
<td>• Multi-tenant support</td>
</tr>
<tr>
<td></td>
<td>• Same code support</td>
</tr>
<tr>
<td></td>
<td>• Customizing support</td>
</tr>
<tr>
<td></td>
<td>• Data sharing support</td>
</tr>
<tr>
<td>Performance</td>
<td>• CPU utilization rate</td>
</tr>
<tr>
<td></td>
<td>• Memory utilization rate</td>
</tr>
<tr>
<td></td>
<td>• Response time</td>
</tr>
<tr>
<td>Application Program</td>
<td>• Received book system</td>
</tr>
<tr>
<td>Function</td>
<td>• Catalog system</td>
</tr>
<tr>
<td></td>
<td>• Inspection system</td>
</tr>
<tr>
<td></td>
<td>• Periodicals</td>
</tr>
<tr>
<td></td>
<td>• Academic journal table of contents system</td>
</tr>
<tr>
<td></td>
<td>• Original document copying service</td>
</tr>
<tr>
<td></td>
<td>• System management</td>
</tr>
<tr>
<td></td>
<td>• DL contents management,</td>
</tr>
<tr>
<td></td>
<td>• Research report management</td>
</tr>
<tr>
<td></td>
<td>• Contents management</td>
</tr>
<tr>
<td></td>
<td>• General user service</td>
</tr>
<tr>
<td></td>
<td>• Original copy management service</td>
</tr>
</tbody>
</table>

4.2.2. Test Method: Table 3 shows the testing method for each test item.

Table 3. TTA Certification Test Method

<table>
<thead>
<tr>
<th>Item</th>
<th>Testing Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>SaaS Maturity</td>
<td>Checking whether the developed items function in conformance with SaaS</td>
</tr>
<tr>
<td></td>
<td>maturity level 3</td>
</tr>
<tr>
<td>Performance</td>
<td>Measurement of server CPU utilization rate, memory usage, and response</td>
</tr>
<tr>
<td></td>
<td>time with 100, 200, and 300 concurrent users, respectively logged in to the</td>
</tr>
<tr>
<td></td>
<td>server under the given test environment</td>
</tr>
<tr>
<td>Application Program</td>
<td>Checking whether the developed items function normally</td>
</tr>
<tr>
<td>Function</td>
<td></td>
</tr>
</tbody>
</table>

The test was performed according to the testing method for each item. The results were evaluated as P (pass), P* (partial pass), F (failure), and NA (non-testable).

4.2.3. Test Result: Table 4 shows the results of the TTA certification test. It indicates that each item of the cloud-based smart library information service system operates normally.
Table 4. TTA Certification Test Result

<table>
<thead>
<tr>
<th>Performance Test Item</th>
<th>Unit</th>
<th>Measurement</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. SaaS Maturity Model</td>
<td>Level</td>
<td>Level 3</td>
</tr>
<tr>
<td>① Connection</td>
<td>Instance</td>
<td>N (customer): 1 (instance)</td>
</tr>
<tr>
<td>② Provided program</td>
<td>Code</td>
<td>Same code</td>
</tr>
<tr>
<td>③ Customizing</td>
<td>Configuration</td>
<td>Configurable by the customer for each tenant</td>
</tr>
<tr>
<td>④ Economy of scale</td>
<td>Yes/No</td>
<td>Yes (instance sharing)</td>
</tr>
<tr>
<td>⑤ Scaling</td>
<td>Yes/no</td>
<td>Replacement to better performing system</td>
</tr>
<tr>
<td>⑥ Data</td>
<td>Format</td>
<td>Data sharing</td>
</tr>
<tr>
<td>2. Performance Test</td>
<td>Class</td>
<td>Class A</td>
</tr>
<tr>
<td>① CPU utilization rate</td>
<td>%</td>
<td>8% or less</td>
</tr>
<tr>
<td>② Memory usage</td>
<td>Byte</td>
<td>500MB or less</td>
</tr>
<tr>
<td>③ Response time</td>
<td>Sec</td>
<td>1 Sec</td>
</tr>
<tr>
<td>④ Portability</td>
<td>Yes/No</td>
<td>Yes</td>
</tr>
</tbody>
</table>

5. Concluding Remarks

The LinkSaaS represents the culmination of a dedicated research program directed at the problems of scientific computing – in particular, that of the field high-energy physics, a field which faces its own imminent data deluge, with experiments generating petabytes of data per year to be analyzed by thousands of highly creative, highly competitive (and yet often highly collaborative) researchers spread across hundreds of institutes around the world. LinkSaaS was thus conceived as an extremely dynamic “global grid”, one that would be able to harness resources beyond the walls of a single institute or scientific community – indeed, to achieve the same global scale as the Internet and the World Wide Web, rapidly changing, cutting-edge code even while assuring resources owners of their security, privacy, and control.

At the same time, it is clear that the scientific research community is hardly the only one with such computing needs: where supercomputing was once the exclusive domain of governments and the largest corporations, a confluence of research, business, and consumer trends have brought it into the mainstream. LinkSaaS brings it even closer: every “edge” device is a potential supercomputer element and front-end, and its straightforward programming model empowers developers to pursue creative network strategies suitable to different problems and scales, rather than being forced into one application model. The examples in this article, spanning several research fields as well as media and general data mining applications, demonstrate the LinkSaaS architecture’s flexibility. Indeed, LinkSaaS encourages its organic expansion in size and application, and as an active open source project with a growing community behind it, its own improvement as well.

Thus, there is a sense in which LinkSaaS completes the revolution begun by the World Wide Web – the last disruptive technology to emerge from the practical challenges of high-energy physics research. As the Web encouraged organic growth of
information dissemination, LinkSaaS now encourages growth of the capacity to consume and integrate it; as the Web encouraged personalized publishing, LinkSaaS now encourages personalized processing. We may be tempted to believe, then, that as the advent of the Web was transformative, challenging, and ripe with opportunities, the advent of LinkSaaS is but the first sight a New World altogether.

The greatest problem concerning library homepages up until now has been that they are not centered on the users but on the library, which is the information provider. As a result, it is not easy even for users familiar with the library homepage to find the desired information on the homepage, which suggests that the use of the various electronic data provided by a library homepage is not that high, particularly because the users still perceive books as the main source of information provided by libraries.

To solve this problem, this paper emphasizes that the new Web 2.0 based library service needs to break away from simply acting as an information manager and play the role of information and knowledge producer. To that end, this paper presents the smart library information service system as a means for libraries to produce and provide information and knowledge centered on the users through its library homepage.

Currently, most library systems use client/server and ASP services for their software. However, such systems are difficult to manage and incur high operating costs because of the problems related to high HW and SW purchasing costs, installation and distribution, customization, upgrade, fault and problem management, and expensive license royalties.

To solve such problems, this study developed the key elements for deploying the cloud-based digital library system in the multi-tenant environment. The system was deployed with the SaaS-based software on-demand type service model, which requires only a small initial investment, is simple and easy to use, and delivers a low cost IT service.

References


Authors

Byung-Won Min

He received M.S degree in computer software from Chungang University, Seoul, Korea in 2005. He received Ph.D. degree on Department of Information Communication Engineering from Mokwon University, Daejeon, Korea. He worked as a professor in the department of computer engineering, Youngdong University, Youngdong, Chungbuk, Korea, from 2005 to 2008. He joined the faculty of the Dept. of Information Communication Engineering, Mokwon University, Daejeon, Korea, in 2008. His main research interests include Ubiquitous Sensor Network and embedded system. Cloud Computing. Smart Library, SaaS