Construction of SaaS-Based e-Learning System in Japan

● Norihiko Sakamoto

(Manuscript received January 30, 2009)

Fujitsu's "Internet Navigware" electronic learning (e-learning) application began to be offered in Japan in the alternative format of software as a service (SaaS) in July 2008 after ten years of life as a packaged product. In this paper, I first outline Fujitsu's objective in converting Internet Navigware to a SaaS application and then describe the functional requirements of SaaS applications for reducing operating costs such as a multi-tenant structure and how those requirements were satisfied in Internet Navigware. I also discuss issues that must be considered, especially in the case of a SaaS application, in the basic design of user rights, session control, and operating commands.

1. Introduction

Fujitsu's electronic learning (e-learning) application "Internet Navigware",[1] which was introduced as a commercial packaged product in 1998 and is still available today, began to be offered in Japan in the alternative format of software as a service (SaaS)[2] in July 2008. Internet Navigware is a training management application providing various capabilities from the development of teaching materials to learning and results management. In addition to e-learning functions, it provides various other functions including group-training management, bulletin boards, and communication functions for question-and-answer corners and other services. Version V9.0 L20 went on sale in August 2008, but work on its conversion to a SaaS application began in early 2008. The early versions called Navigware, V1.0 and V2.0, were configured as a client-server system using original protocol. Version V3.0 marked a transition to a server application based on hypertext transfer protocol (HTTP) to support Internet use and the name was changed to Internet Navigware. Later, in version V8.0, the architecture was restructured to accommodate a large-scale user base on the order of 100 000 users and the program itself was rewritten as a Java application running on a Fujitsu Interstage Application Server (**Figure 1**).

In this paper, I describe the business considerations that prompted Fujitsu to convert Internet Navigware to SaaS and the technical requirements characteristic of SaaS applications, such as a multi-tenant structure, that had to be met in the conversion process.[3]

2. Conversion to SaaS

There were three main reasons for converting Internet Navigware to a SaaS application. The first is that many customers have come to desire services in SaaS and application service provider formats as an alternative to intranet-based services. Since 2005, a number of Fujitsu Group companies have been providing services based on Internet Navigware, and sales in FY2008 jumped by 156% from the previous fiscal year. This rise in sales is attributed to the following customer

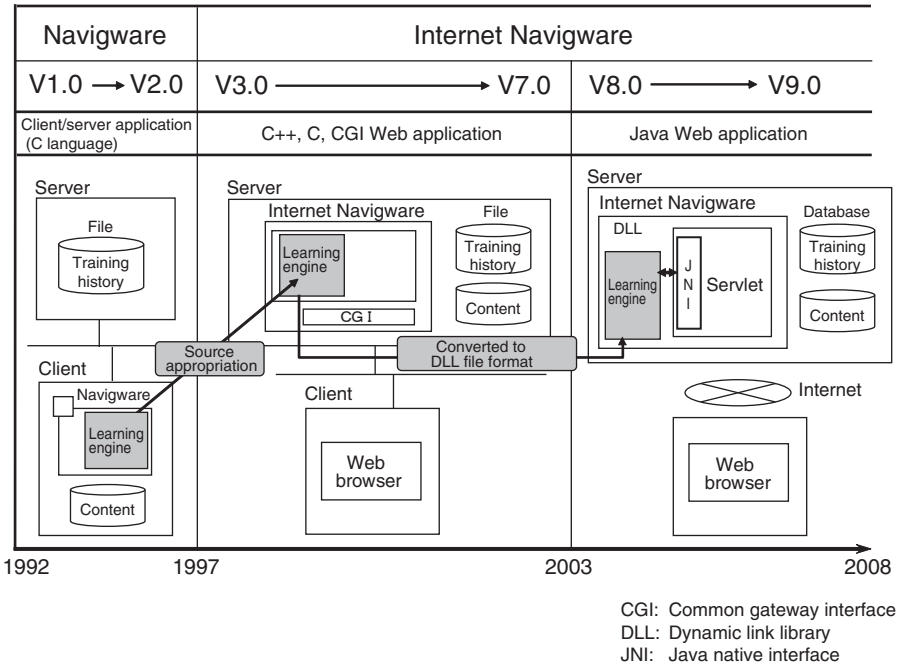

Figure 1
Internet Navigware architecture.

needs and expectations.

- There is no desire to own e-learning assets since e-learning is not part of the company's core systems.
- The use of an e-learning system is desired only when needed. For example, a company may desire to use an e-learning system for security training only periodically, such as twice a year.
- The information systems department has no desire to devote additional energy to an e-learning system and would rather consign its provision to an outside source if at all possible.
- There is no desire to make an investment in an e-learning system without being able to accurately assess its effects in advance.
- Users wish to take lessons in an Internet environment from their homes, branch offices, and other places.

The second reason is Fujitsu's desire to make a transition to a stock type of business. This type of business is attractive because it can provide a stable source of revenue: once a SaaS customer base has been formed, monthly sales can be generated by charging for metered usage.

The third reason is that development resources can be consolidated, which is beneficial for developers. In the case of a packaged product, developers must support various operating systems (OSs) and databases. Specifically, they must support three OS types—Windows, Linux, and Solaris—and three database types—Symfoware, Oracle, and SQL Server. In the case of SaaS, however, they need to support only one OS and one database, which reduces the man hours involved in development and testing. Likewise, program corrections to fix bugs need be made in only one place. There are also benefits in terms of support since support personnel have to be familiar with only the SaaS environment. These benefits can have a great effect on ensuring quality and reducing costs.

3. Functional requirements of SaaS applications

SaaS applications have three main functional requirements. They must have a structure and functions that enable efficient use of resources and reductions in operating costs, they must provide self-customization features for users, and they must have mashup functions. These requirements had to be taken into account in converting Internet Navigware—which was originally designed for use on an intranet—to a SaaS application. Below, I discuss several aspects of a multi-tenant function for making efficient use of resources and reducing operating costs and then describe mashup functions.

3.1 Multi-tenant system

3.1.1 Advantages

The most important concept from the perspective of using server resources efficiently is the multi-tenant system, which allows multiple customer environments to be established on a single server. Although multiple customer environments can actually be established on a single server in a single-tenant system, this has shortcomings. Here, virtual servers may be used, and in the case of an Interstage Application Server, the same application can be initiated multiple times in units of Java processes called work units. Single-tenant and multi-tenant systems are compared in **Figure 2**. In a single-tenant system on one server, there is one application server with multiple applications running on it. The problem with this arrangement is that each work unit represented by a gray box uses memory space independently. Here, each work unit requires at least 512 MB of memory, which means that a maximum of only 4 to 6 work units can be mounted on a 32-bit Windows server. Even if we assume several dozen users per company, the ability to host

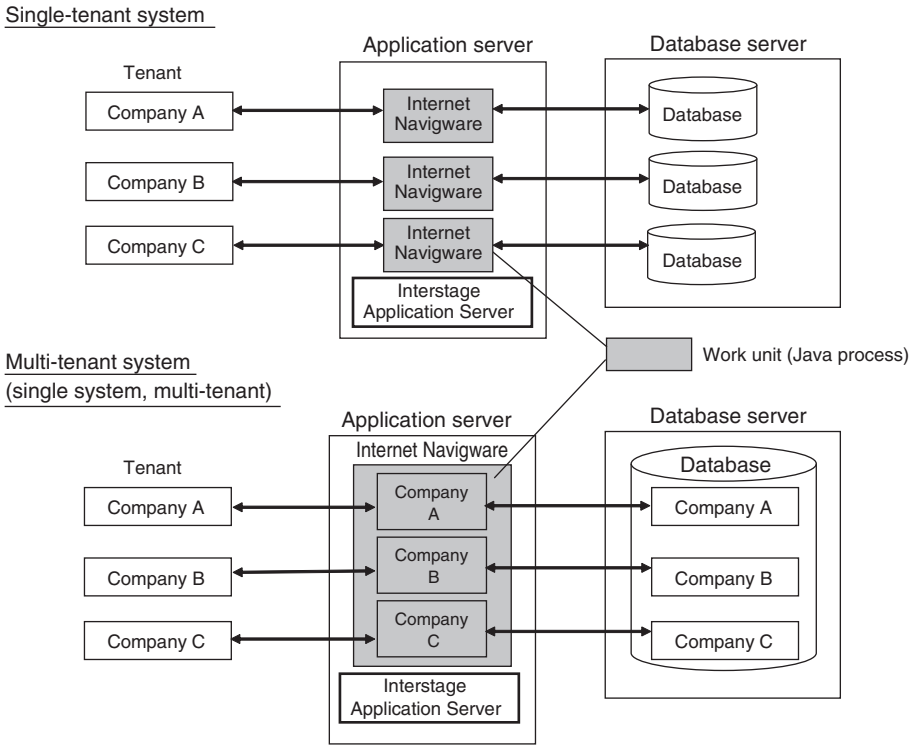

Figure 2
Comparison of single-tenant and multi-tenant systems.

only 4 to 6 companies on a single server does not make for good business from a cost perspective. Moreover, from the viewpoint of operations, a troubleshooting patch would have to be applied to every work unit and backups, operations, and monitoring would have to be performed separately for each unit, all of which increase the load on developers and operators.

In response to these problems, the multi-tenant concept has come to attract attention. In this system, only one application process runs on a single application server. This system can host up to 10 000 users (1000 simultaneous sessions) for any number of tenants if 1 GB of memory is made available to that process. As for databases, while the single-tenant system allocates a logical database to each tenant, the multi-tenant system stores all tenant data in a single logical database. Therefore, the multi-tenant system enables a significant reduction in server resources.

3.1.2 Internet Navigware multi-tenant system

The conversion to SaaS did not require us to implement a completely new version of Internet Navigware. The operating format of having multiple organizations share a server has been a functional requirement of e-learning systems for some time, and even in its packaged-product form, Internet Navigware eventually provided functions that could provide a basis for a multi-tenant system.

The affiliation trees as the foundation for data management in Internet Navigware are shown in **Figure 3**. Here, the first level is "department". Within a department, various types of information are managed on a closed basis. That is to say, the design is such that users in one department can never observe the names of other departments. In addition, the design provides for a "department training manager" in each department who has the right to do anything within that department

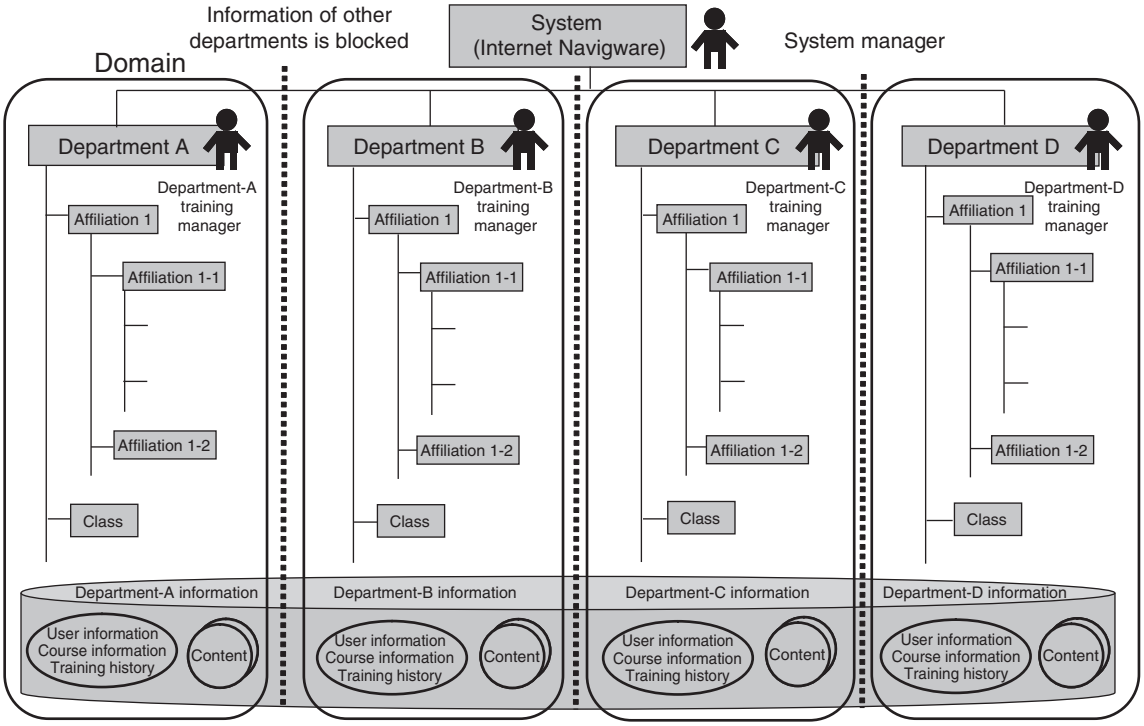

Figure 3
Concept of multi-tenant system for Internet Navigware.

but is prevented from seeing anything in other departments. The reason for this design is a response to customers' requests: organizations with ten thousand or more employees have often requested that different departments be treated like different companies so that one department cannot view the affairs of other departments. This basic structure can be used to support the multi-tenant system of SaaS by using a "department" as a "tenant". Thus, in multi-tenant Internet Navigware, different departments are treated as different companies and the affairs of one's own department are hidden from other departments.

The basic ideas behind our implementation of a multi-tenant system are shown in **Figure 4**. Here, there are four tenants: companies A, B, C, and D. If, for example, Company A's manager registers a user, the server stores the data set consisting of Company A's tenant ID and the user's ID and manages that user accordingly. Similarly, in the case of Company B, the server stores Company B's tenant ID and the user ID of the user in question. This combination of tenant and user IDs can likewise be used to search for user information. Managing data in combination with tenant IDs in this way allows the data of multiple tenants to be managed using a single logical database.

3.1.3 Multi-tenant resource management

While a multi-tenant system does not require as many resources, the fact that data about multiple customers resides in the same process raises issues that do not come up in a single-tenant system. For example, if many users from a certain tenant happen to be logged in at the same time, users from other tenants trying to log in might find operations to be sluggish and the system unusable. In response to such a problem, Internet Navigware allocates each tenant a certain number of simultaneous-connection licenses for the entire server. This limits the number of logins for each tenant.

The manager's screen shown in the upper-right portion of **Figure 5** sets 10 licenses for each tenant. Thus, if Tenant-A users were to log in to the system one after another, the 11th user attempting to log in would be prevented from doing so and would see the message, "The server is busy. Please wait a while and then try again."

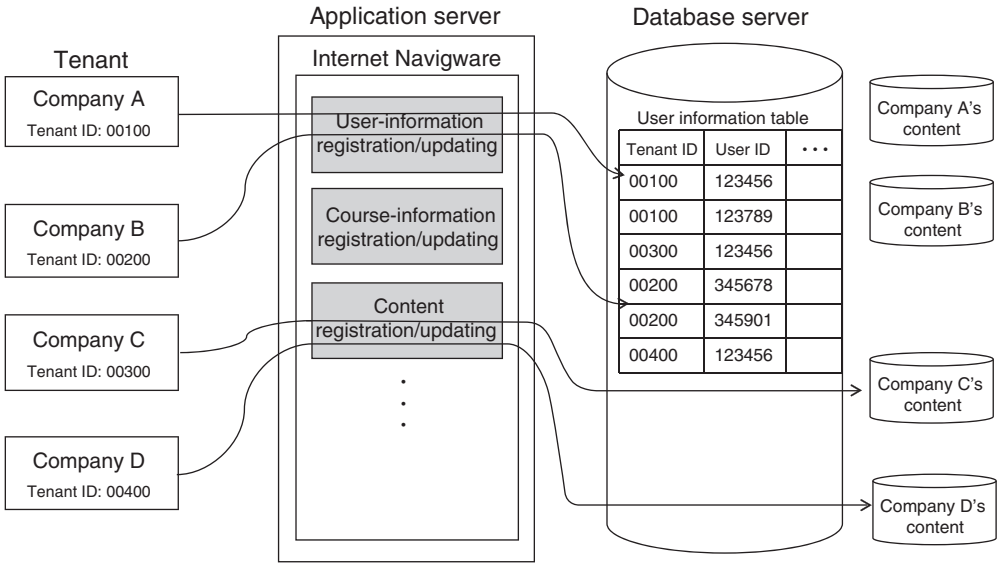

Figure 4
Implementation of multi-tenant system.

(Original is in Japanese)

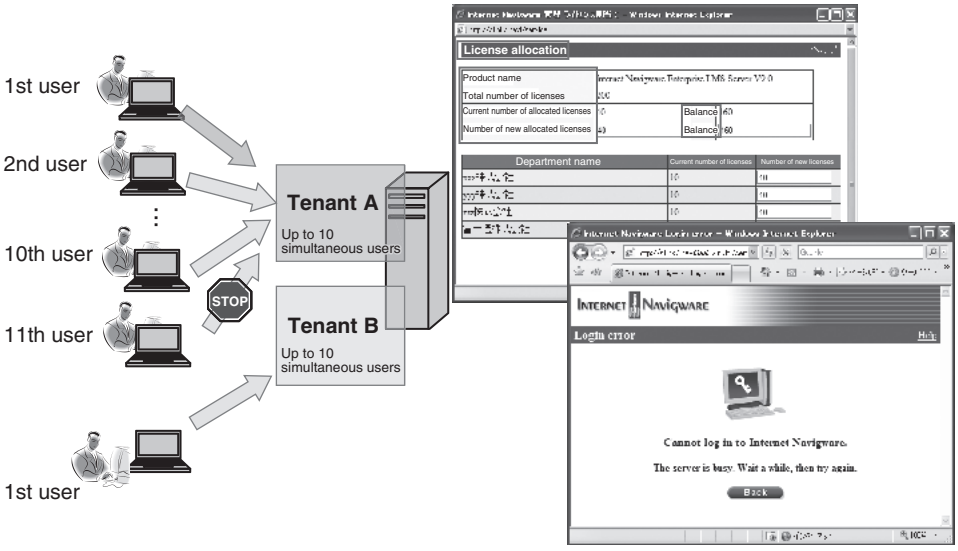

Figure 5
Multi-tenant resource management.

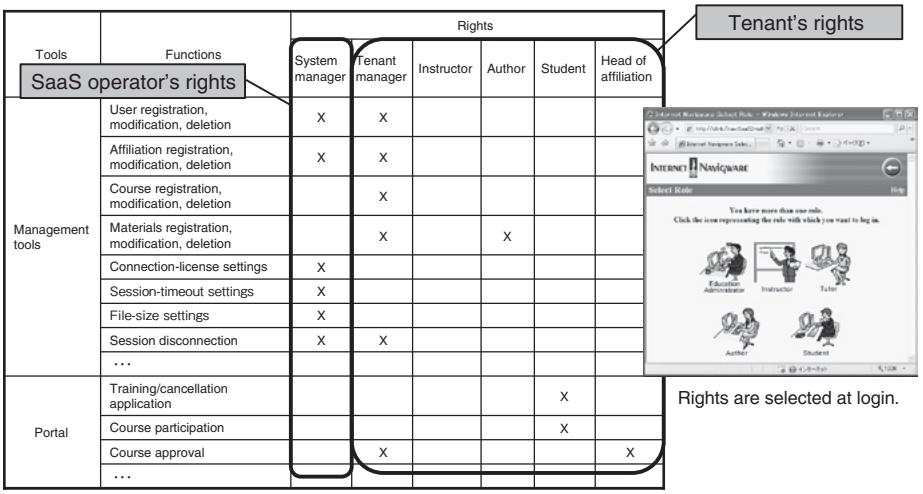

Figure 6
Rights of tenant manager and SaaS operator.

On the other hand, a user of Tenant B attempting to log in would be allowed to do so since only one other user of that tenant is currently logged in. This control mechanism prevents a certain tenant from monopolizing the CPU and database resources. However, this approach still has a problem. A few users of a particular tenant could initiate many operations simultaneously and generate a big load on the database. Therefore, we should consider a mechanism that limits the number of connections to the database from each tenant.

3.1.4 Tenant manager's rights

Next, I describe the design of rights as an effective means of reducing operating costs. The rights and allowed operations in Internet Navigware are shown in **Figure 6**. In the design of system-level rights, it is common to give the system manager (here, the SaaS operator) unlimited rights while users (tenant manager

and students) are given limited rights. In SaaS, however, this kind of design can present problems.

In the case of Internet Navigware, for example, the system manager has the right to set system connection licenses, session-timeout period, and the maximum size of attached files sent to the bulletin board. These rights are deeply related to the management of system resources and are not granted to the tenant manager. Now, let us suppose that the system manager has unlimited rights, including all those granted to the tenant manager, and that the tenant manager is away for some reason. This setup could result in the SaaS operator performing work that is essentially the customer's responsibility because tenant users have no management rights. To avoid this situation, a SaaS application must be designed so as to separate the system-manager rights used by the SaaS operator and the tenant-manager rights used by the tenant users.

3.2 Mashups

3.2.1 Internet Navigware mashup functions

Internet Navigware provides three functions for creating mashups (hybrid Web applications). The first, which is peculiar to Internet Navigware, is described in more detail in Section 3.2.2.

1) Commands in HTTP client form

HTTP-based commands can be used for batch processing.

2) RSS

Really simple syndication (RSS) enables announcements about new courses or system maintenance to be provided to users as Web feeds.

3) SOAP

The simple object access protocol (SOAP) enables Internet Navigware functions to be used in Web services based on Apache Axis.

3.2.2 Commands in HTTP client form

When server products were designed in the past, operation-related commands, such as ones for registering data on the server, were usually designed in a form that involved direct operation on server data or a server application programming interface. In Internet Navigware, however, commands have been implemented in an HTTP client format (**Figure 7**). In addition, communications with a server can be protected

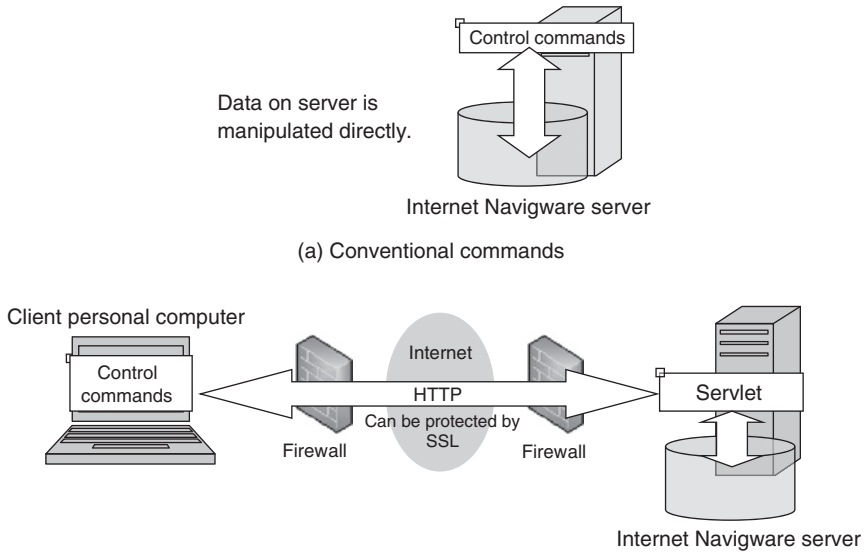

Figure 7
Commands in HTTP client form.

by using the Secure Sockets Layer (SSL) protocol. Since commands are in HTTP form, a connection can be made even without setting a firewall, given an environment accessible by a Web browser. This function enables a customer who wishes to perform batch processing overnight to do so without having to depend on the SaaS operator. It also enables Internet Navigware commands to be combined with an in-house system on customer servers to enable linked processing such as automatic registration of training results in the personnel affairs system.

4. Support for metered charging

SasS is expected to use not a flat rate monthly fee common for Internet access, but "pay-as-you-go" charging, in which usage is metered and customers are later charged for what they actually used. However, many customers would like to set a fixed monthly budget and not exceed it. As shown in **Figure 8**, Internet Navigware has a function for limiting the number of students using the system in a given month of service. If the limit is reached, any new students trying to join a course get an error message and the department training administrator (manager) is notified by E-mail and given the option of increasing the budget and raising the maximum usage limit.

5. Conclusion

The conversion of Internet Navigware to a SaaS application provides the following benefits.

- Reduced operating costs

The multi-tenant structure enables the number of servers to be reduced. Moreover, the number application servers and the amount of middleware for operations and monitoring can also be reduced.

- Reduced development costs

The OSs and databases to be supported can be fixed and the man hours required for development and testing can be reduced.

- Reduced maintenance costs

When program bugs are fixed, only one type of correction has to be made and only one environment has to be tested. In addition, the support team needs to be knowledgeable about only the SaaS environment.

Looking forward, we plan to work on schemes for reducing costs even more and increasing profitability.

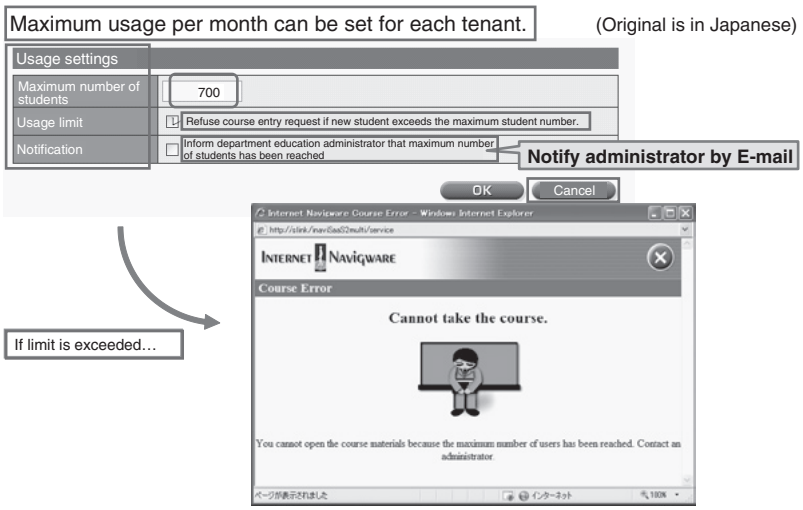

Figure 8
Function for controlling maximum charge.

References

1) Internet Navigware Web site. (in Japanese). *http://jp.fujitsu.com/solutions/elearning/*
2) Internet Navigware as SaaS. (in Japanese). *http://jp.fujitsu.com/solutions/elearning/saas/*
3) Koji Satake: Fujitsu's Activities for SaaS. (in Japanese), *FUJITSU*, Vol. 59, No. 1, pp. 14–19 (2008).

Norihiko Sakamoto
Fujitsu Ltd.
Mr. Sakamoto received a B.S. degree in Secondary Education from Shizuoka University, Japan in 1990. He joined Fujitsu Ltd., Tokyo, Japan in 1990 and has been engaged in the development of Web-based learning software.

E-mail: skmt@jp.fujitsu.com