2008 IEEE International Conference on Services Computing

A Software as a Service with Multi-tenancy Support for an Electronic Contract Management Application

Thomas Kwok, Thao Nguyen and Linh Lam
IBM Research Division
Thomas J. Watson Research Center
19 Skyline Drive, Hawthorne, NY 10532
kwok@us.ibm.com

Abstract

In most commercial electronic contract management applications available today, different customized code base has to be developed, deployed and operated to support each tenant. Few advanced commercial electronic contract management applications use a single code base with configuration options to support multi-tenants. However, a separate instance of the code base still has to be deployed and operated for each tenant even in these applications. The business model of having to support a single application instance for each tenant makes an electronic contract management application and other critical business applications out of reach for most small and medium businesses (SMBs), in particular, the very small businesses (SVBs) because of its high development and maintenance cost. Recently, a new business model of a single application instance supporting multi-tenancy based on Software as a Service (SaaS) has emerged making expensive business applications more affordable for SMBs and SVBs for multi-tenancy [1].

In this paper, we present the first of a kind multi-tenancy SaaS electronic contract management application. We also describe several novel methods used in the metadata, security and shared services, as well as customization and tenant extensions modules to support multi-tenancy SaaS in this application. This multi-tenancy SaaS application has shown to benefit both the application service providers as well as their tenants. This new multi-tenancy SaaS model can reduce the application hosting cost and make the application more affordable to the tenants because of its capabilities in customization and scalability while continuing to support an increasing number of tenants. It furthers benefits tenants by saving their money and time with immediate access to the latest IT innovations and infrastructure improvements on a single application code base. Most end users of tenants have found their productivities increased, the contract transaction time accelerated, contractual errors reduced in using this multi-tenancy SaaS electronic contract management application as demonstrated in several ongoing IBM pilot programs serving more than ten tenants with over 3000 end users.

1. Introduction

Contracts are required in most business transactions in companies of all sizes as they constitute the binding relationships between a company and its suppliers, business partners, or customers. Automation of some contract management tasks in the electronic contract lifecycle creates a substantial value for the company. This value stems from improved productivity and security, effectively aggregated contract information; accelerated contract transaction time and lifecycle processes; reduced contractual errors and risk; enabled revenue forecast and profit optimization as well as better compliance enforcement [2]. Today, there are over twenty available commercial software products on electronic contract management listed in the International Association of Contract and Commercial Managers [3]. However, different companies have different requirements on presentations, business rules, workflows, document flows, database and security in managing their contracts. For the enterprises or large companies, they can provide enough information technology (IT) resources to develop their own electronic contract management applications, or even pay for an application service provider (ASP) or vendor to customize its existing electronic contract management application for them if necessary. For the small and medium businesses (SMBs) and in particular the very small businesses (VSBs), it may not make business sense for them to staff a large IT team to develop their own electronic contract management application, and they cannot afford to pay for an ASP or vendor to customize the application for them.

Recently, a new business model of software applications called Software as a Service (SaaS), which lowers the cost of development, customization, deployment and operation of a software application to support multiple tenants over the Internet has been evolved [1]. In general, SaaS is associated with business software applications. It is a Web-based software application deployed and operated as a hosted service over the Internet and accessed by users. Multi-tenants in addition to multi-users supports, installation of

978-0-7695-3283-7/08 $25.00 © 2008 IEEE
DOI 10.1109/SCC.2008.138

application on a managed Internet data center with remote management accessibility are a few characteristics of a multi-tenancy SaaS application. In the SaaS business model, the ownership, technology infrastructure and management responsibility of the application has moved to ASPs from tenants or customers [4]. It benefits ASPs by reducing hosting cost due to customization and scalability while increasing number of tenants or customers. With a single application code base to support several tenants, the total deployment time is shorter. In addition, updating of application features are simpler and centralized because there is only one instance of a single source code base. It also benefits the tenants or customers through their saving in money and time. The cost to use the application is a per user basis and pays as it goes. The tenants can gain immediate access to the latest IT innovations and improvements provided by the ASP without spending their own IT budgets. However, there are initial setup, configuration and maintenance steps that have to be carried out first in order for the application to support multi-tenants in a SaaS operational structure.

In this paper, we present the first of a kind Web-based electronic contract management application developed from the ground up, deployed and operated over the Internet as a SaaS business model to support multi-tenants. This application allows each tenant to customize the application's presentations and Web pages, business rules, workflows, document flows, data structures, email contents and contract life cycle parameters to meet its requirements through the application's metadata configuration. The following sections in this paper focus on the modules or components of the application specifically designed and developed to support multi-tenancy in a SaaS model.

2. The SaaS maturity model

In a SaaS model, the multi-tenancy support can be applied to four different software layers: the application, middleware, the virtual machine (VM) and the operating system layers. For the application layer, there are four levels of SaaS maturity model as shown in Figure 1 [4]. Level one has a separate instance for each tenant's customized code base and it is similar to the Application Service Provider (ASP) software application model. Level two has a separate instance for each tenant but the instances come from a single code base with configuration options. Level three has a single instance for all tenants with configurable metadata for each tenant. Level four has a load-balanced farm of identical instances with configurable metadata for its tenants. Our multi-tenancy SaaS electronic contract management application is based on the SaaS application layer and uses the third level of SaaS maturity model.

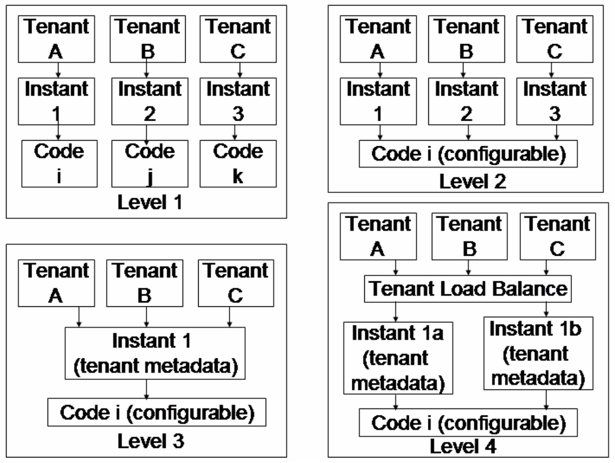

Figure 1. Four levels of SaaS maturity model.

3. The framework of a multi-tenancy SaaS application

The application architecture for the single application instance hosting multi-tenants in the third level of SaaS maturity model requires the implementations of a number of services, such as tenant specific Metadata services, the Security and Shared services, and a SaaS data model. Figure 2 shows an architectural framework of our multi-tenancy SaaS electronic contract management application. This framework provides the capability of hosting multi-tenants using a single or multiple application instances supported by the same set of computing backend servers. It consists of a Web server, a DB2 server and an Application server. The presentation and customization modules of the application are deployed in a Web server. The Security, Directory, Metadata and Shared services, the Tenant extensions module as well as the core modules of the application are deployed in the Application server. The DB2 server contains a central user account database, a central tenant metadata database, and a shared database along with the schema.

First, for each tenant the application provider has to register an administrator and assign a unique tenant identification (ID) using the Security services. Then, the tenant's administrator can create user accounts and register the users. The administrators also register their suppliers, business partners, and their customers for the tenants. The tenant information are stored on a central tenant metadata database through the Metadata services while the end user information are stored on a central user account database through the Directory services. Second, the tenant's administrator has to configure the application metadata in order to customize the application presentations, Web pages, user interfaces and branding, business rules, workflows and document flows, data structures and security policies using the Customization module. These tenant specific metadata are stored in a central tenant metadata database by the metadata services. Third, when end users from different tenants access the application through HTTP or HTTPS communications, the Security and Directory services in the Application server have to authenticate and authorize their accesses to the application. The users are tagged with their tenant ID numbers. The Presentation module in the Web server dynamically creates, composes and delivers customized Web pages completed with inputs and interactions for end users according to their tenant ID numbers using the Metadata services. Fourth, the Shared services in the Application server provide the metering and billing for each tenant. The shared database and schema in the DB2 server is used as a repository for all the electronic contracts and their related documents.

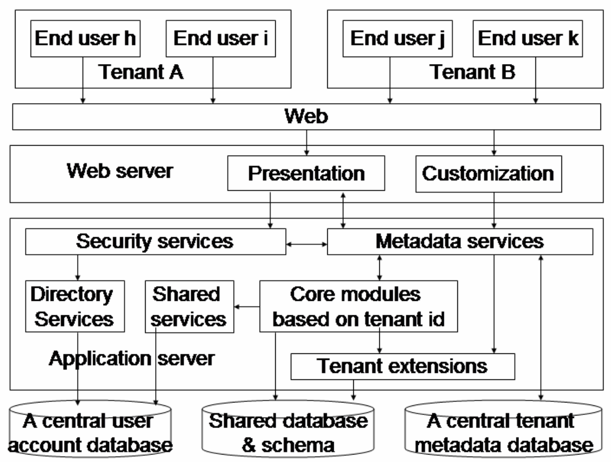

Figure 2. An architecture framework of a multi-tenancy SaaS electronic contract management application.

4. The electronic contract management application

As shown in Figure 3, the main core engine of the application comprises of eight important modules: an Administration module, an Access Control module, a Workflow module, an Email notification module, an Electronic Signature module, a Document Management module, a Data Extraction and Search module, and a Lifecycle Management module. These eight modules are tenant-specific and their functions are driven by the tenant ID numbers. The tagged tenant ID number on each user is passed to all these eight modules by the Security and Metadata services. All of modules interact with the Metadata services to retrieve those tenant specific metadata required from the central tenant metadata database.

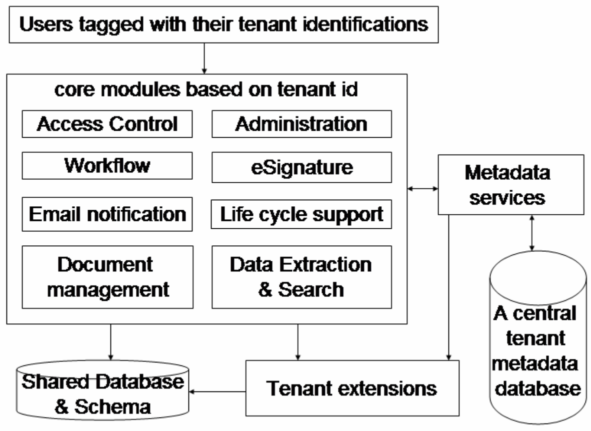

Figure 3. The core modules of a multi-tenancy SaaS electronic contract management application.

The Administration module is used by the SaaS provider or application administrator to setup and configure all system related parameters for each tenant. It is also used by the tenant's administrator to setup and configure all contract related parameters according to the tenant ID number. The Access Control module has to differentiate users in three different categories: the SaaS provider or application administrator, tenant or organization administrators and end users. It also authenticates authorized users to perform and execute certain electronic contract management tasks according to their tenant ID numbers and their roles within the tenant organization. The workflow engine and its configuration tool in the Workflow module supports both internal and intra-enterprises workflows. The Email notification module creates and stores data of the emails ready to be sent out. The content of these emails are tenant specific according to the tenant's metadata through the Metadata services. It has a service daemon to compose the content of the ready to send email and send it out every three to fifteen minutes. It also allows authorized users to track, retrieve and resend emails. The SMTP server acts as an email server to propagate emails to a company's administrators, their suppliers, business partners and customers from the multi-tenancy SaaS electronic contract management application. The Electronic Signature module records and stores electronic contract signing information. It also composes an electronic contract signature file in a viewable form such as a PDF file. Then, it superimposes a watermark of contract signing information on this signed electronic contract document. The formats of signing information can also be tenant specific and are based on the tenant's metadata. The Document Management module uploads, downloads, stores and retrieves all electronic contracts and their related documents. All the electronic contracts and their related documents are stored in the shared database with the same schema using their tenant ID numbers but they are separated and isolated from tenant to tenant. The Data Extraction and Search module extracts metadata from the electronic contract documents. It also provides metadata and keyword search capabilities for users to retrieve their electronic contracts and related documents in the database and repository. Both the search queries and their results are separated according to their tenant ID numbers using the Metadata services. The Life Cycle Management module is a service daemon. It performs a set of pre-defined automated electronic contract management tasks, such as activate signed contracts, cleanup all workflow steps of rejected contracts and delete already expired contracts beyond retention periods. Again, the warning, expiration, archive and retention periods are tenant specific. These parameters are stored or retrieved as metadata in the central tenant metadata database by the Metadata services. More detail descriptions of functions in these eight modules in an electronic contract management application have been published elsewhere [5, 6].

All communications among contract administrators, their suppliers, business partners and customers go through one or more secure wired or wireless networks. Similarly, the interactions and connections between the Web server, the application server and the remote or local databases are also carried out through one or more secure wired or wireless networks. As a result, this multi-tenant SaaS electronic contract management application enables authorized users to perform different contract document tasks and sign electronic contracts in real-time simply using a Web browser without additional required hardware and software installation. Many contract representatives, each assigned with similar or different roles can simultaneously access this electronic contract system using a Web browser 24x7. A service provider, such as IBM, can host this multi-tenancy electronic contract management application as SaaS to a number of tenants to manage their electronic contracts using the same set of computing servers and resulting in lowering the cost and simplifying system integrations.

4. Tenant Specific Metadata Customization

In the SaaS environment, the application hosted by the application provider is shared among multiple tenants on a common application platform. However, the application is usually parametric to allow a wide range of customization within a basic set of functions; this allows the tenants not only to have their preferred look-and-feel but also to have a unique user experience. The basic set of customization that our multi-tenancy SaaS electronic contract management application provides for tenants is captured in the Metadata Customization module as shown in Figure 4. The Metadata Customization module enables each tenant to configure the GUI presentation of Web pages, the configuration of the templates for the document workflow and the business process flow.

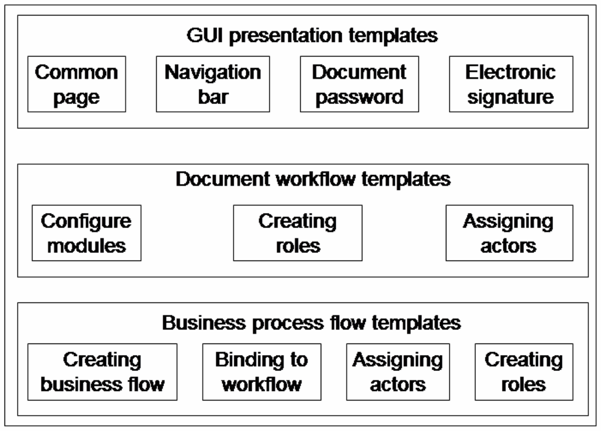

Figure 4. The Metadata Customization module for multi-tenants.

The application presentation data common to the displayed Web page segments such as the master header, footer, left and right navigation bar, popup master header and popup footer are all configurable. Such customization can be, for example, loading the popup master header with each tenant's company logo, providing different items in the navigation bar and various sizes for contract text area. The left and the right navigation bar can be configured to use the contract terminologies familiar or specific to each tenant group. Further, all the images used for the application presentation can be configured to load from different sources provided by their location paths. In addition, some of the metadata can also be configurable, for example, the password to be used for securing documents for each group of tenants. Each group of tenants can provide a predetermined password to be added to the primary contract document(s) when the contract reaches completion so that it can no longer be modified. This document password is also used to unlock an uploaded contract document which is password protected to allow further contract processing for the group. If there are more than one security passwords available, the system will try to apply these passwords according to their preconfigured priority. Another important customization is the electronic signature used to sign a contract. Tenants have different requirements on what legally accepted by their country and corporation to be a valid electronic signature when signing contracts online. As a result, our multi-tenancy SaaS electronic contract management application provides each tenant to configure what to use for its electronic signature as well as the legal verbiage that accompany the signing of the contracts. The accepted legal signature can be one of the followings – electronic password, electronic signing pad or hardware assisted signature device.

Our multi-tenancy SaaS electronic contract management application allows the system administrator from each tenant to configure the contract document flows to fit his or her internal contracting requirements. The document flows are a set of templates supporting different contract modules such as Negotiation, Request for Quote, Purchase Orders etc. The administrator can decide which of the modules to include in the tenant's contracting portfolio. Once the modules have been selected, the administrator can then configure the templates for these modules. The template of a module consists of a default sequence of steps for which the administrator needs to create the roles for these steps and then assign specific user or a group of users to perform those roles. Our multi-tenancy SaaS electronic contract management application also allows the system administrator from each tenant to further configure business flow templates, which can be configured to reflect the tenant's proprietary workflows into the contract document flows. These business workflows can be associated or bound to any step of a contract document flow to support any special requirement from a tenant's organization. For example, if the tenant's organization requires its electronic contracts over certain monetary value to go through multi-levels approval process, then the business flow implementing such an approval process can be created and associated to an appropriate step in the contracting document workflow. For each step of the business flow, the administrator also needs to create the roles and assign specific user or a group of users to perform those roles.

5. Multi-tenancy SaaS Security Services

In a multi-tenancy SaaS model, the security services consist of authentication and authorization. The application provider has to authenticate an administrator of each tenant. Then, the application provider delegates the authentication of end users from each tenant to its administrator. The multi-tenancy SaaS authentication can be implemented using three alternatives [7]. They can be implemented using a centralized authentication system, a decentralized authentication system or a hybrid approach authentication system. Our multi-tenancy SaaS electronic contract management application uses a centralized authentication system and it is shown in Figure 5. This authentication infrastructure is simple, easy to design and implement because it requires no change to the tenant's own user authentication infrastructure. However, it has some difficulties when implementing with a single sign-on authentication policy. Thus, end users are required to login manually more than once.

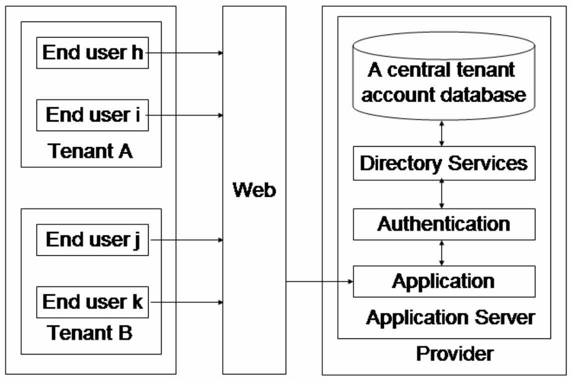

Figure 5. A multi-tenancy SaaS centralized authentication system.

Authorization assigns roles to individual user according to their tenant IDs, groups and accounts. It gives one or more permissions to the user to perform specific operations or actions on the application. A single permission can be assigned to one or several roles. Each user will be granted the union of permissions assigned to all roles to which the user belongs. In a multi-tenancy SaaS model, there are three different types of access: access for the application provider, access for the tenant's administrator and access for the end users. Business rules can be used to further control access to actions and resources than permissions allow. Access control can be modified, added or deleted as appropriate. A default set of roles, permissions and business rules are available to all tenants in this application. This application also allows an individual tenant to customize these default rules or create new additional rules.

In a centralized authentication system, an application provider manages a central database for its tenants and their user accounts. Our multi-tenancy SaaS electronic contact management application uses tenant ID number and a password to authenticate its tenants and the end users. Before a new tenant can use the application, the administrator of the application host provider needs to create a unique tenant ID number and register an administrator for this tenant. An administrator can be a coordinator or some one who can take up the responsibility for registering their users. When registering a new user, the coordinator inputs the user's general information, selects a role for the user and assigns the user an access level. There are different roles that users can act on the electronic contracts – for example, submitter, approver, reviewer, signer and counter-signer. Each role allows the users to perform certain tasks when the contracts are routed to them. In addition to the role assignment, the coordinator also assigns users an access level to further authorize the users' access to the contracts within their organization. There are four access levels that a coordinator can choose from to assign to a user. First level access entitles users to access any electronic contracts within their organization. Second level access entitles users to access electronic contracts belonged to their department and other contracts of other departments with special authorization within the organization. Third level access entitles users to only access those electronic contracts they have submitted. The user role and the access level along with the user's general information form the user's profile, which is used for the user authentication and authorization whenever the user logs into the system or performs any contracting tasks.

6. A multi-tenancy SaaS data model

In a multi-tenancy SaaS model, a default data model, such as a standard database with default tables, fields, queries and relationships is not flexible or secure enough to separate and isolate each tenant's data. Ideally, a centralized multi-tenancy SaaS data model should have network-based access, low overhead, and should be robust and secure. Tenant's specific data and queries should be separated, isolated and secured from each other. There are three approaches to managing multi-tenant data in a SaaS model discussed in [8]. The first approach is a separate and dedicated tenant database. The second approach is a shared database and schema with a fixed extension set and/or custom extensions. The third approach is a shard database with separate schemas or custom extensions.

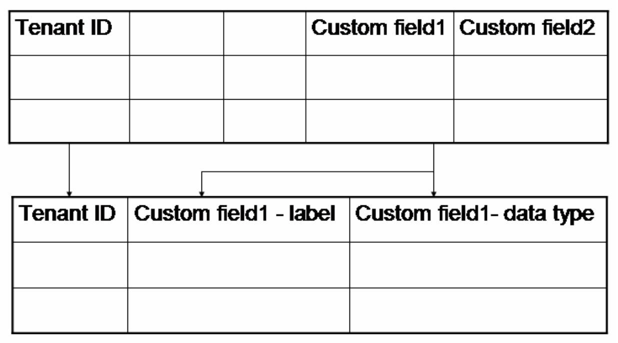

Figure 6. A shared database and schema with fixed extension set.

Our multi-tenancy SaaS application uses the third approach to manage multi-tenant data - a shared database and schema with fixed extension set as shown in Figure 6. Since a multi-tenancy SaaS application shares computing resources and software codes among all the tenants on the same set of servers, the data of the tenants are also hosted in the same database with the same set of tables in this third approach. This approach uses a tenant ID in a table column to associate every record with the appropriate tenant. Each record has a number of pre-allocated custom fields (typed or un-typed) that tenants can use for their own purpose. The tenant specific data is stored in a fixed extension set with these pre-allocated fields. The hardware and backup cost of this approach is the lowest. However, the extensibility of the data model is limited to the number of custom fields provided. It is also harder to restore a tenant's data from backup. It may incur additional development effort in security.

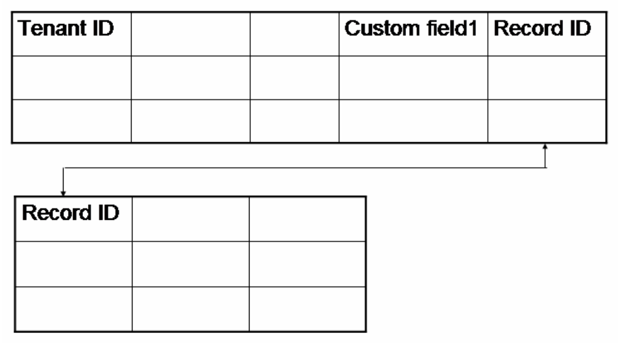

Figure 7. A shared database and schema with custom extensions.

In addition to the above approach, our multi-tenancy SaaS electronic contract management application also uses an enhanced approach to manage multi-tenant data - a shared database and schema with custom extensions as shown in Figure 7. In this enhanced approach, the tenant specific data is stored as name-value pairs in a separate table. The tenant record with tenant data is assigned a unique record ID that matches one or more rows in a separate extension table. It allows tenants to extend the data model arbitrarily. A data type identifier is needed in the third column, as data in the tenant specific data table cannot be typed. However, this enhanced approach has an added level of complexity for database functions, such as searching, indexing, query, and updating records.

7. Multi-tenancy SaaS shared services

There are several shared services that a multi-tenancy SaaS application needs to implement to facilitate the system [9]. A metering and billing system to accurately track tenants' usage of the application, and bill them for the time and/or resources used accordingly. A system to monitor site access and application performance to ensure that service level agreements (SLAs) are being met for all tenants. A method to restrict or throttle access at certain times of the day in order to meet the certain criteria of each tenant's SLA. Operation support services such as account activation, provisioning and service assurance may also be needed. Business support services such as billing and tenant management are also required.

There are different pricing models that a multi-tenancy SaaS application provider may use to charge the tenants for using the hosted application. Some providers may charge tenants on a per-seat basis, a mix of per-seat charge with a volume-based surcharge, a transaction-based fee or a flat monthly fee. Our multi-tenancy SaaS electronic contract management application uses utilization metrics to allow charging tenants according to their usage of the application. The metrics capture the following usage information: service provider; organization; group; user; module; operation; total size of contract documents; operation time and response time. These metrics allow metering the usage of the application at a finer granularity than just the total utilization. They enable metering of the application by service provider, individual organization and user. The metrics support distribution of operation charges among users, groups and organizations that use the application.

When a user accesses the application, the information such as which service provider, organization and group, to which the user belongs, are recorded into a log file or database. The contract module type and the operation being accessed are also recorded. The module types can be one of the followings: Contract, Request for Quote, Request for Price, Quote, Negotiation, Purchase Order, Proposal, Archive or Administration. Since the module type is recorded, it allows the service provider to charge different usage rate for each module type if necessary. The operations can be one of the followings: Transaction, Search, Administration, Upload, or Download operation. A Transaction operation is one that when an action is performed on a contract it causes the contract status to change. These operations can be reviewing, approving, signing or counter-signing a contract. A Search operation is an operation where a user performs a search query for any contract lookup. All the administrative operations such as registering a user, configuring a workflow or generating a statistic report are recorded down by their activity name in addition to the operation name. Metering application usage at the level of tracking the operations accessed allows the hosting provider of the electronic contract management application to have a billing model to charge tenant for exactly what they use.

Unlike other operations mentioned before, an upload operation is metered differently. An operation is viewed as an upload operation when a user submits or updates a contract. A download operation, on the other hand, is one when a user clicks on a contract document link to download the contract content for viewing. When recording an upload or download operation, the total byte size (in kilobytes) of the contract document is also recorded. It allows the hosting provider to charge this operation according to the storage capacity or the database utilization. For all operations, beside the operation name, the operation time and the response time are also recorded. The operation time timestamps when an operation is performed can be used in application usage reports or application auditing trails in a later date. The response time records the total elapsed time from when the server receives an operation request until when the server completes the operation. The response time can be used to monitor and validate the application performance against what is being advertised in the SLA by the hosting service provider.

8. Implementation and Industrial Experience

Most of the features and functions of this multi-tenancy SaaS electronic management application described in this paper have been implemented. This application also supports six phases of a contract life cycle: request for proposal, quote, negotiation, execution, active and archive contract. This application has been used by the IBM Global Service Division in several IBM pilot programs. As this electronic contract application supports both internal and intra-enterprises workflows as well as the multi-tenants hosting capability, IBM can host the application as a SaaS to a number of tenants or companies at the same time in each pilot program. As a result, the cost of providing the electronic contracting services to multiple tenants or several companies is minimized by sharing a set of computing servers like a number of IBM Websphere middleware products.

Our industrial experience has confirmed the stability and usefulness of this multi-tenancy SaaS electronic contract management application based on the feedbacks from the tenants and their users. It has shown to reduce cycle time and costs of contract transactions because of the Email Notification service. It enhances productivity of sales and support teams in managing their electronic contracts. It saves a lot of time for the contract users, increases their efficiencies and productivities by simplifying and automating many contract tasks, such as merging several documents into one along with new page numbering and contract life cycle management. It has also shown to improve customer satisfaction by using a standardized format on superimposing the electronic signing information as a watermark on the signed electronic contract document. It minimizes the amount of security exposures of electronic contract documents because the required manual steps for the contract users are replaced with an automatically tasks. It also creates new opportunities for driving incremental revenue from the Search and Data Mining module. Above all, a satisfactory latency or response time is well maintained for an application instance serving over ten tenants with more than 3000 end users.

9. Conclusion and Discussion

This multi-tenancy SaaS electronic contract management application enables the electronic contract users to accelerate the transaction time and life cycle of their electronic contracts with the Email Notification service. It enhances the productivity of the contract users by simplifying and automating some manual and tedious contract life cycle management tasks. This system also provides an automated and efficient way for a user to superimpose the electronic signing information as a watermark on a signed electronic contract document in a secure environment. Moreover, the system substantially eliminates the potential for human errors and security breaches in the electronic contract process task and the on-line signing document tasks. This application has been deployed in a number of pilots with multiple IBM business units, brands and channels as different tenants. These pilots presently have more than 500 external customers (business partners and suppliers) and around 3000 contract users. Nearly all of them have found their productivities improved, the electronic contract transaction cycle accelerated, and contractual errors reduced. The pilot results have confirmed the anticipated benefits, such as reduced time and costs, improved ease of doing business, better tracking and increased business process control, etc. Above all, it has proved the concept of one application instance supporting multiple tenants using the same set of servers.

The acceptance of the electronic contract comes mainly from its abilities to reduce expenses, improve turn around time, and enable collaborative tasks with business partners in a secure environment. Our multi-tenancy SaaS electronic contract management application has enabled multiple tenants to execute and manage their contracts entirely online with very low cost. It has shown to reduce the time it takes to complete contracts from days to minutes, driving down processing costs and increasing productivity with anytime, anywhere access to contracts. The key attributes for success for a multi-tenancy SaaS application are its configurability for customization, its multi-tenant efficiency and its scalability. In this paper, we have emphasized both the customization and multi-tenancy but we have not addressed the scalability. In the future, we should investigate different techniques for scalability, such as scaled up by moving the application to a larger, more powerful server, scaled out by running the application on more servers and scaling the data by partitioning the data into smaller segments in order to improve the efficiency of queries and updates or by replication. In addition, database security and data isolation, such as permissions, trusted database connections, secure database tables, tenant view filter and data encryption, remain as the main challenges for a multi-tenancy SaaS application.

10. Acknowledgment

The authors would like to acknowledge the implementation works done by K. Roy, A. Shah, M. Dharmarajan, S. Maradugu and T. Chieu. The authors would also like to thank A. Mohindra and J. Batstone for their supports.

11. References

[1] C. Pinhanez, "Service Systems as Customer-Intensive Systems and Its Implications for Service Science and Engineering", *in Proc. of the 41st Annual Hawaii Int'l Conference on System Sciences,* pages 117 - 117, 2008.

[2] W.M. McGovern and L. Lawrence, Contracts and Sales: Cases and Problems, Matthew Bender, 1986.

[3] International Association of Contract and Commercial Managers, http://www.iaccm.com

[4] Iod, Sotware as a Service, Kogan Page Ltd, 2002.

[5] T. Kwok T. Nguyen, L. Lam and T. Chieu, A Web-based and Email Driven Electronic Contract Management System, in *Proc. of the 3rd IEEE Int'l Conf. on e-Business Engineering,* IEEE Computer Society, pages 149-156, 2007.

[6] T. Kwok and T. Nguyen, An Automatic Method to Extract Data from an Electronic Contract Composed of a Number of Documents in PDF Format, in *Proc. of IEEE Int'l Conf. on e-Technology, e-Commerce and e-Service,* IEEE Computer Society, pages 258-262, 2006.

[7] K. Bennett, et al., "Service-Based Software: The Future for Flexible Software, Software Engineering Conference," *in Proc. of 7th Asia-Pacific APSEC,* pages 214-221, 2000.

[8] A. Sehai and W. Felix, Utility Computing, Springer, 2004.

[9] G. Bunker, Delivering Utility Computing: Business-driven IT Optimization, Wiley, 2006.