Configurability in SaaS for an Electronic Contract Management Application

PRADEEP KUMAR ARYA
II M.E CSE
Anna University Coimbatore,
Academic Campus
pradeeparya25@gmail.com

V. VENKATESAKUMAR
Lect/CSE Dept
Anna University Coimbatore,
Academic Campus
mail2venkatesa@yahoo.com

S. PALANISWAMI
Registrar
Anna University Coimbatore,
Academic Campus
joegct81@yahoo.com

ABSTRACT
In the last few years Software as a Service (SaaS) has been accepted as a core concept. SaaS is a software delivery paradigm where the software is hosted off-premise and delivered via web to a large number of tenants and the mode of payment follows a subscription model. Configurability allows the single instance multiple tenant model which leads to many benefits both for the customers and the vendors which in turn has led to the acceptance and popularity of SaaS. Configurability in SaaS software aims to provide tenants/customers with a multitude of options and variations using a single code base, such that it is possible for each tenant to have a unique software configuration. Few advanced commercial electronic contract management applications use a single code base with configuration options to support multi-tenants Software as a Service (SaaS) has emerged making expensive business applications more affordable for small and medium businesses (SMB) and very small businesses (SVB) for multi-tenancy.

The paper aims to provide information on the nature of configurability in SaaS software, how it can be provided and the technologies needed to support it. 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 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

Keywords
Software as a Service, SaaS, Configurability,

I. INTRODUCTION
Internet is the recent years has advanced in terms of speed, connectivity and reliability for the customers, who could not attract the massive market directly, SaaS has been a curiosity for the market of small customer. A typical SaaS provider offered applications specifically designed to be hosted and delivered over the Internet to many customers. With SaaS, the customers' IT people had access to a number of different features or capabilities within the software set and could fine-tune them via a Web interface to fit the needs of their company. Since SaaS applications were centrally managed/administered by the vendor, there was no need for in-house application developers or experts at customers end. Maintaining a single code-base was simpler and cheaper for the SaaS vendor which in turn led to faster upgrades and cost savings at the customer end.

Due to the faster upgrade cycle of SaaS software and small deployment time, the customers need not spend time and money in upgrades and deployment of the software. On the vendors' side, not only the vendors could tap newer markets through SaaS [5] Contracts are required in most business transactions in companies of all sizes as they constitute the binding relationship between a company and its supplier, business partners, or customer. Automatic of some contract management task in the electronic contract lifecycle create 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 [27]. 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.

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 [7]. It is a Web-based software application deployed and operated as a hosted service over the Internet and accessed by users. In the SaaS business model, the ownership, technology infrastructure and management responsibility of the application has moved to ASPs from tenants or customers [15]. 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.

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.

1.1 Technology enablers for saas
SaaS has established itself as a successful method to deliver business services across the Web. [20] Lists the market drivers for SaaS which arise mainly due to the economic benefits of SaaS. The market drivers for SaaS have existed over 9 years.

• Web 2.0 refers to business, social and technology evolutions. Socially this has led to the internet establishing itself as the communication platform per se and the browser becoming the universal user interface. Technologically the ability to read and write against dynamic, near real-time data and information and
to program against remote functionality to create a new class of web applications leveraging those capabilities has enabled SaaS to emerge.

- **Rich Internet applications** (RIAs) are web applications that have the features and functionality of traditional desktop applications [4]. The concepts of SaaS and RIA are tightly linked: the potential success of SaaS naturally relies on the market's potential to produce good web applications: people will not be willing to give up their traditional software unless the web-based alternatives prove to be equally good [26].

- **Service-oriented architecture** (SOA) has established its value in enterprise systems as architecture for integration, consolidation, and reuse [13]. If software is supplied via SaaS using an SOA approach, companies can employ SOA ESBs between external SaaS applications and internal line-of-business applications.

- **Cloud computing** can further leverage the economic benefits of SaaS for the vendors by not investing in the hosting infrastructure and utilizing the cloud services to offer SaaS solutions. Virtualization is the key behind cloud computing.

- **Virtualization** is an abstraction of computer resources. Wikipedia states that "It creates an external interface that hides an underlying implementation." Virtualization is a means to maximize the use of existing hardware, operating system or application resources because it can be used to simulate a resource that is not physically there.

### II. CONFIGURABILITY IN SAAS

The single instance multiple tenants model of SaaS software requires that every SaaS vendor provide support for configurability. Though most of the organizations today have standard software requirements like CRM, ERP etc., each of these organizations have some unique needs regarding user interface, data, business processes and rules, etc. The SaaS software’s architecture should support configurability without which it would be impossible to use the single instance of the software for different customers.

#### 2.1 Designer and user at tenants

The configuration data is created and used by each tenant. Not all users in the tenant could be allowed to create configuration data for the tenant. This leads to the fact that there need to be two different kinds of users of any SaaS application – designer and user. The designer configures the SaaS application which essentially means setting up some config (configurability) data for the users in his organization.

The designer interface is used by the tenant’s designer to configure (edit) the SaaS application and the user interface is used by the tenant’s users to run (play) the configured SaaS application.

#### 2.2 Configuration data and application data

From the discussion in the earlier section, it follows that designer creates the config data and the user consumes the config data for each SaaS application. The config data plays the most important part in realizing configurability in SaaS application using the same code base and different config data.

#### 2.3 Configurable aspects of SaaS software

There can be different aspects of SaaS software which can be configured. The following sub-sections explore configurability in different aspects of SaaS software.

- **2.3.1 User Interface**

  Configurability of user interface means the ability to change the look and feel of the UI available to the users. A customer can configure the user interface to change its look and feel or to reflect corporate branding. The UI features like icons, colors, fonts, titles, etc. can be changed. There could be option to change the style of the control using skins. To allow for data binding there needs to be option for adding variables. The events could be predefined or could be custom made combining the already existing events as a workflow.

- **2.3.2 Workflow**

  Other than configuring the UI to give unique look and feel to the SaaS applications, it is important that the designer be able to configure the behavior of the application. This is important because it is possible that the same kind of workflow may have different behaviors in different organization. Workflows can be either for collaboration among different users or to compose bigger operations out of the existing operations. A facility to build the meta-model of the workflow needs to be available in the designer part of the SaaS software. This is similar to configuring UI.

- **2.3.3 Data**

  Data is at the heart of any SaaS application. Data drives the SaaS applications. In case of a particular type of SaaS application, that is, in a specific domain like banking, there will be similar type of data across different tenants. Thus data configurability is needed to cater for extensibility in the data model as required by the tenants.

- **2.3.4 Access control**

  Each tenant using the SaaS software will have multiple individuals using the software. The responsibility for creating individual accounts for end users, and for determining which resources and functions each user should be allowed to access lies with the tenant. Users will be grouped into different roles according to the organizational structure and the access control privileges can be configured for each of these. The access control privileges specify what data and UI (forms) an entity in a particular role can access. Hence the designer module must cater for building both roles and users to be able to configure access control for a tenant.

  While catering for configurability, the designer module for any SaaS software must be able to deal with multiple levels/scopes of configuration units which might be required by the tenants. Since any organization is generally organized into
a hierarchical structure, these scopes are also hierarchical in nature. Configurability options should be available to each tenant to set the scopes as required by the tenant.

2.3.5 Other configurability concerns

There can be more configurability options like domain specific extensions in case of SaaS software in the horizontal like customer relationship management (CRM) in case of tenants in diverse domains like IT services, insurance, etc. There could be configurability options for multiple locales for setting up language, date, time, etc which could be a requirement of tenants whose offices are geographically distributed across the globe. What and how much configurability to provide in the SaaS software depends on the targeted customers’ needs. The user experience while configuring a SaaS application plays an important role in success/acceptance of any SaaS software.

III. SUPPORTING CONFIGURABILITY

3.1 Data storage to support configurability

Representation of config data is an important consideration when designing for configurability in SaaS application. A desirable feature in the format of config data would be extensibility. One of the ways to represent config data could be using xml which allows for extensibility.

XML and relational databases are two different technology sets which are very different from each other. XML supports hierarchical data model, whereas databases support relational data model. It has to be taken care of as file I/O by the application through XML parser. Relational databases provide transaction and rollback support which is not present in case of XML.

There are two requirements for SaaS software with respect to data storage - one is for storing config data and the other is for storing app data. Unlike app data which follows normally relational data model used in an enterprise, config data is generally hierarchical in nature. The access control data is also hierarchical in the sense that each tenant may have one or more apps. Each app has one or more roles and each role has one or more users. The other points to consider when deciding on the storage of configure data and app data is that unlike app data which is generally huge for enterprises, config data is small in size. Config data for app data (which may be relational or hierarchical, but small and less frequently updated like other types of config data) may be stored in XML or RDBMS format or may not be required to be stored explicitly if the RDBMS provides access to the metadata.

3.2 Supporting configurable aspects of SaaS software

UI Configurability could be supported by allowing for changes in the UI through templates comprising of style sheets and skins. To provide better configurability of user interface, a GUI builder could be provided to the designer.

The workflows are stored in workflow specific config data formats. Whenever a workflow is triggered, its data is fetched and it is uploaded in the specific user’s login. The data is isolated for different tenants by using one of the schemes for storing app data. Unlike app data which follows normally hierarchical in nature, configurability options should be available to each tenant to set the scopes as required by the tenant.

IV THE SAAS MASTERY 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 [15]. 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.
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 [2]. 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.

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. 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. 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. 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.

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. 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. 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.

VII. 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. Our multi-tenancy SaaS electronic contract management application uses a centralized authentication system and it is shown in Figure 5[2]. 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.

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. 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. This application also allows an individual tenant to customize these default rules or create new additional rules. Our multi-tenancy SaaS electronic contract 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. 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. 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 belonging 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.

IX. 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.

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 shared database with separate schemas or custom extensions.

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[2]. 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. 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.

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[2]. 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. 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.
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[2]. 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.

X. 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 system to monitor site access and application performance to ensure that service level agreements (SLAs) are being met for all tenants. Operation support services such as account activation, provisioning and service assurance may also be needed. 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. The metrics capture the following usage information: service provider; organization; group; user; module; operation; total size of contract documents; operation time and response time. The metrics support distribution of operation charges among users, groups and organizations that use the application. The contract module type and the operation being accessed are also recorded. The module types can be one of the following: 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.

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. 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 can be used to monitor and validate the application performance against what is being advertised in the SLA by the hosting service provider.

XI. IMPLEMENTATION

Most of the features and functions of this multitentancy SaaS electronic management application described in this paper have been implemented. This application also supports request for proposal, quote, negotiation, execution. 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.

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 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 also creates new opportunities for driving incremental revenue from the Search and Data Mining module.

XII. CONCLUSION

Configurability in SaaS enables customers to implement a system which can be used in a very short amount of time, bypassing on premise challenges such as server provisioning and software installation. The keys to “infinitely conceivable” are standards and decomposed services. The standards provide automated service-oriented ways in which composite solutions come together with less work, and decomposed services ensure there is useful content to draw from.

Configurability allows for some unique features in each tenant’s application. But a tenant should not expect everything unique in the application. SaaS is one of the biggest technology trends to affect business applications in recent years. Due to evolving market place and emerging technologies, SaaS architecture details and implementation details are open to research.

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. 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. 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. 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.

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. 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. In this paper, we have emphasized both the customization and multitenancy but we have not addressed the scalability. 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.

XIII. REFERENCES

[2] 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


[6] Configurability in SaaS (Software as a Service) Applications Nitu MCSI, MIETE Scientist ‘D’ ISSA/DRDO, Metcalfe House, Delhi, India. Telephone No:+91-11-23882158. nitu2805@gmail.com


