

Software as a Service: Configuration and Customization Perspectives

Wei Sun, Xin Zhang, Chang Jie Guo, Pei Sun, Hui Su
 IBM China Research Lab, Beijing 100094, P.R. China
 {weisun, zxin, guocj, sunpei, suhui}@cn.ibm.com

Abstract

Software as a Service(SaaS) provides software application vendors a Web based delivery model to serve big amount of clients with multi-tenancy based infrastructure and application sharing architecture so as to get great benefit from the economy of scale. Though SaaS application is usually developed with highly standardized software functionalities to serve as many clients as possible, many clients still ask for function variants according to their unique business needs through easy configuration and customization. Due to the subscription based model, SaaS vendors need take a well designed strategy to enable self serve configuration and customization by their customers without changing the SaaS application source code for any individual customer. In this paper, we will explore the configuration and customization issues and challenges to SaaS vendors, clarify the difference between configuration and customization. A competency model and a methodology framework have been developed to help SaaS vendors to plan and evaluate their capabilities and strategies for service configuration and customization.

1. Introduction

Software as a Service(SaaS) is gaining momentum with the significant increased number of vendors moving into this space and the recent success of a bunch of leading players on the market[1][2]. Designed to leverage the benefits brought by economy of scale, SaaS is about delivering software functionalities to a big group of clients over Web with one single instance of software application running on top of a multi-tenancy platform[3]. Clients usually don't need to purchase the software license and install the software package in their local computing environment. They use the credentials issued by the SaaS vendor to log onto and consume the SaaS service over Web through an Internet browser at any time and from any where with Internet connections. As well known, in the enterprise software area, software is designed to support a business' operations in terms of business data management, process automation and optimization,

and governance. SaaS is not an exceptional case though it does provide major differentiating advantages around much lower Total Cost of Ownership(TCO) and stronger mobility allowed for the users through the Web based delivery model[4]. However, every client is unique which leads into the requirements' variance to the software. The fundamental causes of the requirements variance among clients include: industry focus differences; customer behavior differences; product offering differences; regulation differences; culture differences and operation strategy differences. Therefore most of enterprise software applications need to be tailored more or less so as to effectively serve a specific client. The widely used approaches of tailoring software are configuration and customization. Software configuration and customization is not a brand new problem. There are many academic research and industrial best practices available already, for example: Software Configuration Management(SCM) theory was developed by Roger Pressman through software engineering research[5]; SAP software applications have strong configuration and customization capabilities through Graphic User Interface(GUI) based tool and script based programming tool(ABAP)[6]. The leading SaaS vendors have developed profound configuration and customization capabilities as well, for example, Salesforce.com provides Apex to facilitate the extensive application configuration and customization on the Web based on the multi-tenancy architecture.[7] As well understand, the delivery model of SaaS does not allow the SaaS vendor to develop and maintain a version of application code for each individual customer, the configuration and customization capabilities of SaaS service play an extremely important role for its success. However there is not a thorough study of the configuration and customization issues of SaaS from both business and technology perspectives to guide a SaaS vendor to plan and execute strategies around configuration and customization. In the section 2 of the paper, we explore the difference between configuration and customization, as well as the challenges in the SaaS multi-tenancy environment; In section 3, we introduce the configuration and customization competency model and also the analysis of the aspects which are usually demanded

for configuration and customization; In section 4, we present a methodology framework to guide SaaS vendor to plan and execute configuration and customization strategies.

2. Configuration and Customization in Multi-tenancy Environment

The fundamental design point of SaaS is to serve hundreds and thousands of clients through one instance of software application. SaaS vendors don't develop and keep different software versions for any individual client. The most ideal case for SaaS vendors is that every client feel comfortable using a completely standardize offering. However this ideal case usually doesn't happen in enterprise software application area. As illustrated in figure 1, In general with the functional complexity increase of the software, the more potential tailoring efforts need be involved to serve a specific client. Web e-Mail is one SaaS service with relatively simple functions, that is why clients usually just need to tailor the service with parameters based setting, e.g. e-mail box storage size; account number. Industry generic Customer Relationship Management(CRM) is one service with medium level of function complexity, that is why you see many CRM SaaS vendors offer much stronger tailoring capabilities through configuration and customization tools. As SaaS leverages economy of scale of clients' number through a long tail play, therefore the more complex of the software, it becomes more non-appropriate to explore SaaS model as client may ask very complex tailoring requirements which can not be handled effectively with Web based delivery model in multi-tenancy environment. That is why the most successful SaaS services stay at CRM, Human Resource Management(HRM). Finance & Administration (F&A) and Collaboration(email, web-conference, etc) spaces.[8]

Tailoring a SaaS service can leverage two major approaches: Configuration and Customization. These two terms usually get people confused about their differences. Actually different SaaS vendors use different terms in different contexts. We try to clarify the difference between them. As depicted in figure 2, In order to make a standardized SaaS offering to serve a specific client, we need to tailor it into a tenantized offering by satisfying this client's unique requirements.

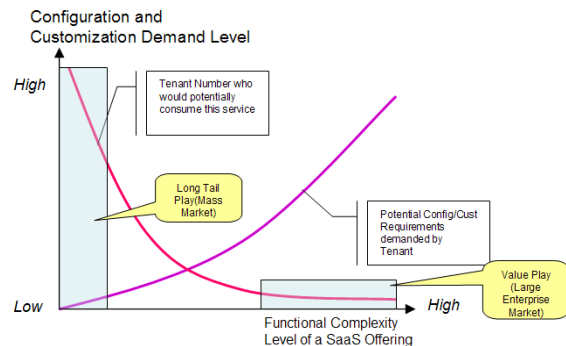


Figure 1. Configuration and Customization Demands vs. Functional Complexity of a SaaS

Both Configuration and Customization can support this tailoring effort into certain level. The crux of the difference is complexity. Configuration does not involve source code change of the SaaS application. It usually support variance through setting pre-defined parameters, or leveraging tools to change application functions within pre-defined scope, e.g. adding data fields, changing field names, modifying drop-down lists, adding buttons, and changing business rules, etc. Configuration can support tailoring requirements within pre-defined configurable limit. Customization involves SaaS application source code changes to create functionality that is beyond the configurable limit.

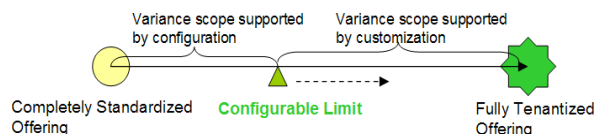


Figure 2. Configuration and Customization

Comparing with Configuration, Customization is a much more costly approach for both SaaS vendors and clients. As Customization involves SaaS software source code changes, that produces many issues which involve significant cost, for example: Requiring people with higher skills with higher wage to work on Customization; Allocating resources and infrastructure to manage software code versions; involving much longer lifecycle brought by code development/debugging/testing and deployment; and losing business opportunity from those clients who can not accept the Customization complexity and cost[9]. The Customization is becoming much more complex in SaaS context, as SaaS vendors need to maintain every piece of Customization code tenant by tenant. Upgrading the SaaS application should not lead into losing of any single tenant's customization code. Therefore wherever possible, SaaS should avoid Customization by using Configuration to meet


clients’ tailoring requirements and enlarge configurable limit as far as possible toward client’s unique requirements.

3. Configuration and Customization Competency Model

To facilitate strategy definition and execution discussion around SaaS configuration and customization, we introduce the Configuration and Customization Competency Model described in Table 2. There are 5 levels of competencies have been defined from “Entry”, “Aware”, “Capable” levels to “Mature” and “World Class” levels. This model can be used in the assessment of SaaS application to identify improvement goals around configuration and customization through necessary

benchmarking with market leader’s competency level. Different level of competency can enable different level of variance requirements through different technical approaches supported by ranges of SaaS services from completely standardized offering across all the tenants to fully tenantized offering for any individual tenant. In theory, the higher of the competency level, the more customers and the more complex variance requirements the SaaS service can support. However different SaaS vendors may have different strategies in terms of targeted customer segments, supported scope of variance, etc. If the SaaS strategy is well defined, the SaaS service can succeed on the market even its configuration and customization competency only stays at “Entry” level or “Aware” level.

Table 2. SaaS Configuration Competency Model

Level of Competency	Description	Approach	Variance Level Supported	
Entry	Highly standardized offering without any configuration and customization support	Well design the functionalities as standardized offering to cover targeted customers	None	 <p>Completely Standardized Offering</p> <p>Fully Tenantized Offering</p>
Aware	Relatively standardized offering with pre-defined variance points	Offer parameterized configuration	Low	
Capable	Relatively standardized offering with user defined configuration	Offer self serve configuration tool to empower customers	Medium	
Mature	Base offering with programable enviroment to enable user preferred customization	Offer scripting based programming for very flexible customization	High	
World Class	Offer a platform supported by programming model and tools to enable extremely strong customization or even new application development	Offer well defined programming model and tools to enable extensive customization and new application development	Extremely High	

The configuration and customization to a SaaS application can happen in many different perspectives. Summit Strategies Inc analyzed the configuration and customization capabilities of SaaS in different implementation layers of the software, which include: Presentation Logic, Application Logic, and Database Logic[10]. Here we try to analyze this issue from clients’ requirements point of view as follows.

As illustrated by the figure 3, SaaS tenants can potentially have configuration and customization requirements from many different perspectives. Each tenant may raise the following challenges to the SaaS vendor: “I need more fields to describe my business documents”; “Our manager wants a new report/dashboard to analyze sales data”; “Our organization has no role of procurement manager”; “The workflow of our business is different with what you can support”. Any of these challenges can be divided into implications to different perspectives of the SaaS

application, e.g., Data, User Interface, Organization Structure, Processing Logic, Workflow, Business Rules, For example: When tenant wants to change the default data structures provided by the SaaS vendor, the configuration and customization tools should support

“Add Custom Field”, “Change Field Name”, “Change Field Type”; When tenant wants to change workflow pre-built by SaaS vendor, the configuration and customization tools should support “Switch on/off Tasks”, “Add New Tasks”, “Reorder the Tasks”, “Change Roles for a Task”. If you analyze those change impact relationship” lines on the figure, you will notice “Data Configuration and Customization” and “Organization Structure Configuration and Customization” are the two most important perspectives, any change of these two perspectives will potentially bring major impact to many other perspectives including User Interface, Workflow, Business Rules, etc. For example: If tenant changes a data structure, then the user interface to support the input and view of the data

should be changed as well; If tenant changes the roles' definition, then the workflow need to be changed accordingly for those tasks handled by those roles. Therefore SaaS vendors should put much more effort to well design Data and Organization Structure layers to

support easy configuration and customization. It is also very important to consider the impacts and establish the linkages between the other software artifacts and the changes of Data and Organization structure.

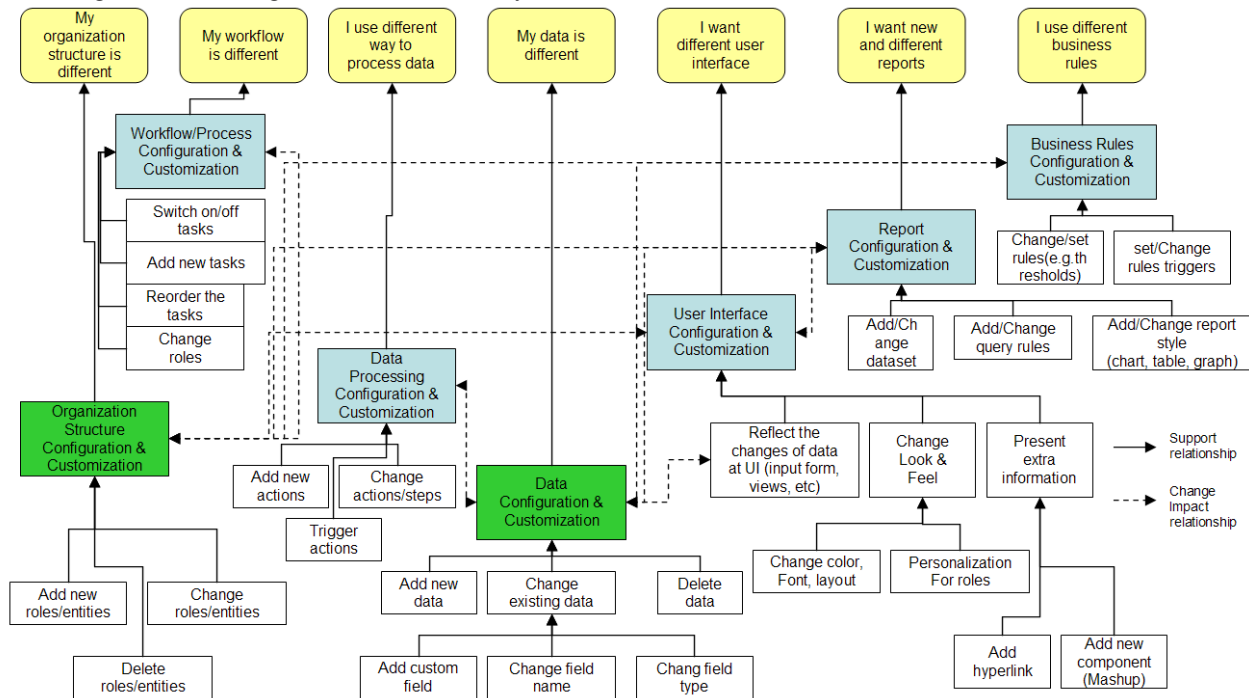


Figure 3 Perspectives of Configuration and Customization Requirements

4. A Framework to Plan and Execute Configuration and Customization Strategy

It is very important to define the appropriate software functional scope to be offered as SaaS. It is extremely critical for SaaS vendors to have the right strategy and software architecture to support Configuration and Customization. This is the foundation to support a SaaS service to pursue economical scale.

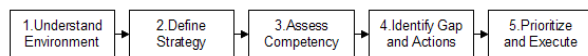


Figure 4. A Framework to Plan and Execute Configuration and Customization Strategy

As illustrated in the above figure, we introduce a framework to guide the plan and execution of SaaS configuration and customization strategy. This framework consists of a methodology and supporting analysis tools.

“Understand Environment”: The first step of the methodology is to make necessary investigation to

understand the environment related with the configuration and customization of the SaaS service. There are two main areas need to be investigated: client requirements and market leader competency level. The objective of analyzing customer requirements is to identify the targeted customer segment and the required variance scope. As illustrated in figure 5, a customer segmentation analysis can be conducted by segmenting the whole market into 4 quadrants divided by two major dimensions: uniqueness of requirements and capability to acquire alternative solution. In general, the customers in quadrant III should be the primary targeted customer segment of SaaS service as these customers have relatively low level of variance requirements and has relatively weak capability to acquire alternative solution (e.g. invest on a custom developed application) other than the SaaS service. Quadrant I is usually a difficult segment for SaaS service to win as each of the customers in this segment has very unique requirements and they have the capability to explore other alternatives.

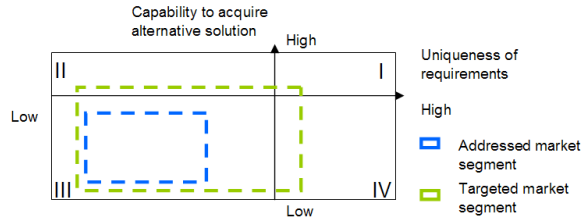


Figure 5. Customer Segment Analysis

Customers in quadrant II and IV are the segments where customers are usually in marginal position. If the SaaS vendors could offer strong, easy and low cost configuration and customization capabilities, they can win more customers in these two segments. The SaaS vendor can leverage survey and interview with selected potential customers to develop such a customer segment analysis. There are two important elements of this analysis. The first one is to well define the SaaS application function scope and complexity level so as to make clients fall into quadrant III as many as possible;

the second one is to determine the addressed market segment and targeted segment through enhanced configuration and customization according to the investigated variance requirements.

Market leader's competency level investigation can help SaaS vendor to well position itself in the competition environment from configuration and customization perspective, which is an important exercise to support target competency level definition discussed later in this paper.

“Define Strategy”: SaaS vendor should determine how they plan to support the required configuration and customization requirements in the targeted customer segment. To facilitate the discussion, we abstract the potential strategies around Configuration and Customization into four Models illustrated in the table below.

Table 1. Different Configuration and Customization Approaches

	Model A: Native Design	Model B: Smooth Evolvment	Model C: Pulse Evolvment	Model D: Failure Management
Description	Thoroughly analyze the common configuration and customization requirements before building the SaaS application; Design the application for extensive configuration and customization; provide powerful web based tools for tenants to configure and customize the SaaS service by themselves or other system integrator vendor.	SaaS vendors need to spend effort to support every single tenant's configuration and customization requirements. But SaaS vendors have a way to manage the cost by leveraging tools & assets, and gradually reduce the cost spend on configuration and customization per tenant.	SaaS vendors collect configuration and customization requirements from a group of tenants. Upgrade application to satisfy the requirements demanded by a big group of tenants when the return on investment can be justified by potential benefits brought by the effort.	SaaS vendors support configuration and customization for individual tenant. They fail to manage the cost within a scope required by a profitable business.
Approach	Provide programming model, web based tools and API for tenants to conduct configuration and customization in self-service mode. SaaS vendors won't change application code for any individual tenant.	SaaS vendors change application codes according to tenants' requirements. They deploy management tool and process to manage the cost spent on each tenant.	SaaS vendors change application codes when the configuration and customization requirements are defined and justified by a big group of tenants' demand.	SaaS vendors change application codes according to each tenant's requirements. They don't have effective tools and process to manage the cost spent on each tenant.
Possible Scalability	Very High	Medium-to-Low	Medium-to-High	Very Low
Application Complexity	Medium-to-Low	Medium	Medium	High

As shown in figure 6, the four approaches, “Native Design”(Model A), “Smooth Evolvment”(Model B), “Pulse Evolvment”(Model C), “Failure

Management”(Model D), have different level of impact on the SaaS service delivery cost spent on each tenant from configuration and customization. As shown on the

following figure, Model D is obviously a bad one which every SaaS vendors should avoid to get into. The other three models can all support sustainable SaaS service business with different profit margins. They can be good choice according to specific SaaS business context in terms of: application complexity, scalability target, the vendor's understanding of the market, the budget situation, etc. In general, Model B is more appropriate for SaaS targeting very limited number of clients as supporting each individual tenant's unique requirements is a very expensive strategy. If a SaaS vendors want to explore very high scalability to leverage very big economical scale (Long tail play), then Model A would be the best approach. The easiest approach for a SaaS vendor starts from is Model C, they learn the market along the process and eventually can be evolved into Model A when they clearly define and build configuration and customization capabilities needed by the large amount of tenants they want to acquire and serve.

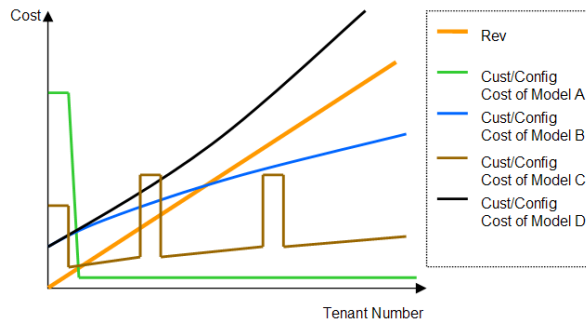


Figure 6. The Impact on Configuration and Customization Cost of Different Approaches

Model A can only be built out through deep understanding of the potential configuration and customization requirements associated with the SaaS service. It takes specially designed software architecture and provides web based tools for easy and extensive configuration and customization without changing the SaaS application source code.

“Assess Competency”: This step is extremely important for those traditional software application vendors who plan to explore SaaS as a new delivery model. The configuration and customization might not be a big challenge for applications vendors who have been successfully addressing consumer market and small medium business(SMB) market. These vendors usually take volume play model and do not support individual customer's variance requirements. They can jump start to explore SaaS by transforming their applications into multi-tenancy enabled with Web interface. However the configuration and customization issue is a big challenge for those application vendors who have been addressing medium to large enterprise

market. Though they have well packaged application as a base, these vendors are usually paid by their customers to take custom development approach to satisfy each individual customer's unique variance requirements. In many cases, source code level customizations are involved if the application has no well defined configuration framework. But in the traditional application delivery model, the vendor can afford that because they are paid by the end customer to do so. This approach can not be replicated in SaaS delivery model. SaaS has subscription based usage pattern. The very small amount upfront investment made by the tenant and monthly based subscription fee can not support the total cost spent on source code level customization. Therefore these application vendors should be very careful and make necessary assessment about their competency around configuration and customization before they decide to move their application to the SaaS delivery model.

We introduced the configuration and customization competency model with several major perspectives to be studied for a SaaS application in section 3. These perspectives can be categorized into 6 groups: data structure and processing, organization structure, user interface, workflow, business rule and reporting. This model can be used to assess the competency of the existing software application from configuration and customization aspect. As illustrated by an example in figure 7, a benchmark study can also be conducted to compare with market leader's competency so as to clearly identify competency improvement goals. This study does not mean that every SaaS vendor should improve the competency to the higher level from every perspectives. If a vendor's application is pretty similar with other existing SaaS services on the market from function aspect, then higher level configuration and customization competency can help the vendor to get stronger competitive advantages.

“Identify Gaps and Actions”: Through the competency benchmark study, the competency gaps can be identified to guide the actions' definition. The example in figure 7 shows the gaps and improvement goals especially in user interface and reporting perspectives. With the analysis in section 3, the competency improvement goal could be further developed into specific actions. For example, to improve the configuration and customization capability for reporting function of the application, the actions need to be identified to support *“Add/Change dataset”*, *“Add/Change query rules”* and *“Add/Change report style (chart, table, graph)”*. If we go through every configuration and customization perspectives according to figure 3, we can generate a long list of actions which implies very complex design challenges for the SaaS

application. It is critical to have well designed principals and approach to tackle all the actions in a unified and consistent way.

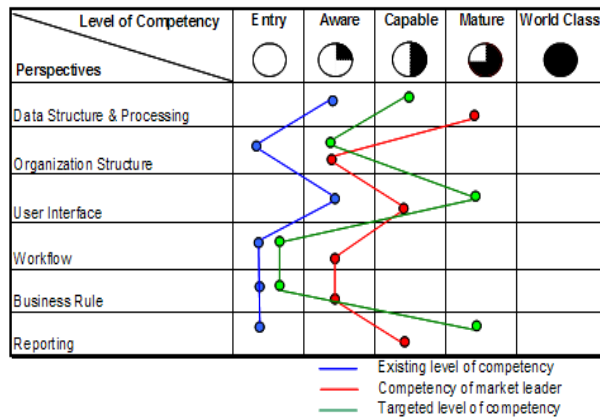


Figure 7. Competency Assessment and Gap Analysis

As we discussed in table 2, there are different approaches which can enable different competency level requirements. Parameterized Configuration can enable “Aware” level variance through setting pre-defined parameters and options by end user in the runtime environment. Self Serve Configuration tool leverages an application variance metadata framework and a series of simple point-and-click wizards, users can design custom user interfaces and modify the structure of the data model and the application’s business logic (workflow, business rules, etc). But the scope of configuration is constrained by the metadata framework. Scripting based programming, a version of end-user programming, allow for larger scope customization by the end user by extending the features of the tool using a constraint scripting language to guarantee security and avoid script generated damage to the core application. World Class SaaS service make their application coupled with a development environment and a formal programming model that can be used by user to build new application code or modify compiled code to match their requirements. Different SaaS vendors take different approach and develop their own implementations.[11] There is not a generic and platform independent approach for SaaS vendors to use today. There is a strong opportunity for research activities.

In this section, we introduced a framework to guide SaaS vendors to plan and execute configuration and customization strategy. This framework comes from our hands on experience and lesson learned through developing SaaS application and business. The framework can be tailored to help SaaS customers to evaluate different SaaS vendors to make the subscription decision.

5. Summary

Configuration and customization are the critical perspectives for SaaS vendors to design their offerings. Though the most important thing is to well design the SaaS application functions to satisfy as many customers’ requirements as possible in the targeted customer segment and application domain, the competency level of configuration and competency implies the key competitive advantages on the market. In many cases, if the SaaS application is very difficult to be designed as standardized offering for most clients, strong configuration and customization capability will become the key success factor. In this paper, we clarified the concepts of configuration and customization in SaaS context, and introduced competency model and a framework to help SaaS vendors to plan their offerings from configuration and customization enablement point of view. In the future, we will explore a runtime and development framework which offers programming model and environment to enable SaaS vendors to quickly build highly configurable and customizable SaaS applications.

References

- [1] E. Knorr, “Software as a Service: The Next Big Thing”, http://www.infoworld.com/article/06/03/20/76103_12FEsaas_1.html.
- [2] Wei Sun, etc, “Software as Service: An Integration Perspective”, ICSOC 2007.
- [3] Chang jie Guo, etc, “A Framework for Native Multi-Tenancy Application Development and Management”, CEC/EEE 2007.
- [4] Sheryl Kingstone, “Understanding Total Cost of Ownership of a Hosted vs. Premises-Based CRM Solution”, Yankee Group Report June 2004.
- [5] Roger Pressman, “Software Engineering: A Practitioner’s Approach”, McGraw-Hill Science.
- [6] SAP, “Getting Started: ABAP, SAP Community Network
- [7] Salesforce.com, “Introduction to the Apex Platform”
- [8] Summit Strategy, “Software-Powered Services Net-Native Software-as-Services Transforms the ISV Business Model”
- [9] Clotilde Rohleder, “SOFTWARE CUSTOMIZATION WITH XML”, Volume VI, No. 2, 2005, Issues in Information Systems.
- [10] Summit Strategy, “Software Powered Services Net Native SaS Transforms The Enterprise Application”
- [11] Datamonitor ComputerWire, “SaaS Customization”