Journal of Computer Science & Computational Mathematics, Volume 8, Issue 3, September 2018
DOI: 10.20967/jcscm.2018.03.003

The Five Ws Taxonomy on Customization of Software as a Service Applications

Abdulrazzaq Qasem Ali*, Abu Bakar Md Sultan, Abdul Azim Abd Ghani, Hazura Zulzalil

Dept. of Software Engineering and Information System,
Faculty of Computer Science and Information Technology, Universiti Putra Malaysia
*Corresponding author email: s100032@yahoo.com

***Abstract*:** Software as a Service has become a common model for various business applications. Increasingly, software application development companies have followed this model in delivering their software to their customers. SaaS delivery model is a multi-tenant model which reduces operational cost by increasing the resource sharing of one instance of application among multi customers (commonly named tenants). In this model, SaaS provider is not allowed to provide a copy of application code for each tenant, yet they have to permit each tenant to fulfill their different and varying requirements by enabling them to customize the application. This paper explores SaaS customization and the processes which make up the overall procedure. Accordingly, the Five Ws (who, when, what, where, and why) method is used to capture the core elements of SaaS customization, as well as how customization plays a role for the various SaaS stakeholders to introduce a taxonomy of SaaS customization. This taxonomy could provide guidance on SaaS customization.

***Keywords*:** software as a service, customization, multitenant, five Ws, taxonomy.

1. Introduction

The current most economically viable method of distributing software is to make a generic application which is relevant to a large number of customers, which is known as Software-as-a-Service (SaaS) [1]. In this instance, the SaaS provider is unable to provide a unique SaaS application for each customer, and so this is where a change needs to be made. To address the distinctive needs of each customer in terms of function and quality, a SaaS application must be customizable. Additionally, a multi-tenancy application is reliant upon this ability to be customized and successful [2]. An example of what may be customized is the QoS requirements.

Therefore, when building customizable SaaS, it is vital to capture all aspects of SaaS customization to guide and facilitate the customization process. This paper explores SaaS customization and the processes which make up the overall procedure. Accordingly, the Five Ws (who, when, what, where, and why) method is used to capture the core elements of SaaS customization, as well as how customization plays a role for the various SaaS stakeholders. By answering these questions and organizing the results, the taxonomy of SaaS customization is structured. This taxonomy could provide guidance on SaaS customization and lead to adequate decision making with regards to the SaaS development.

In this paper, Section 2 summarizes the methodology of this study. Section 3 shows the formulation of each question, and it is expected outputs. Section 4 reports the answers for each question. Section 5 offers conclusions, limitation and suggestions for future work.

2. Study Method

The Five Ws was used as a means to address the main elements in SaaS customization, as shown in Figure 1. Each of the Ws was discussed and illustrated by using previous researches about Software and SaaS customization to capture the core elements of SaaS customization, and then to map among these elements to introduce a taxonomy of SaaS customization.

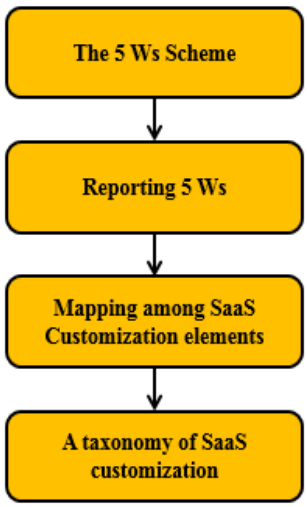

Figure 1. Method

3. The Five Ws Scheme

When customizing multi-tenant SaaS applications, one is faced with more than just issues related to different pieces of data or GUI. Beyond that, there are still the actors, the devices, time and locations, and many more factors involved in the experience which ought to be taken into consideration in SaaS customization. These components form the minimal set of issues that need to be addressed in SaaS customization. The Five Ws (who, when, what, where, why) are used to break down the wealth of information available, and their proposed results are outlined in Table 1.

3.1 Who

It is imperative to identify who is involved directly in SaaS customization. This deals with the number of people involved and their roles. For different situations, there may be a different number of people participating in that experience. By answering the 'who' question, we can understand who is qualified to perform the customizations in a given situation.

3.2 What

As any number of the elements of a SaaS application can be customized, it is important to make clear to which a report is referring at a given time.

3.3 When

Knowing when customization should take place means being able to implement the variability space involved with SaaS applications ahead of time and predicting the right kind of customization to be prepared automatically for future unique requirements.

3.4 Where

This addresses the physical locations of SaaS application architecture that must be involved in the customization process.

3.5 Why

It deals with the reasons why SaaS customization matters. Thus, answering these questions enables us to understand the benefits of having a customizable SaaS application.

Table 1. The Five Ws Schema

W	Question Form	Expected Outputs
Who	Who is directly involved in SaaS customization?	Main players in SaaS customization
What	What to customize?	Assets of SaaS application to be customized
When	When should the customization take place?	Customization within SaaS DLC
Where	Where should the customization take place?	SaaS customization layers
Why	Why SaaS customization?	Benefits & Motivations

4. Reporting Five Ws

Using the Five Ws, this paper introduces a taxonomy of SaaS customization concepts: see Figure 2 for an outline.

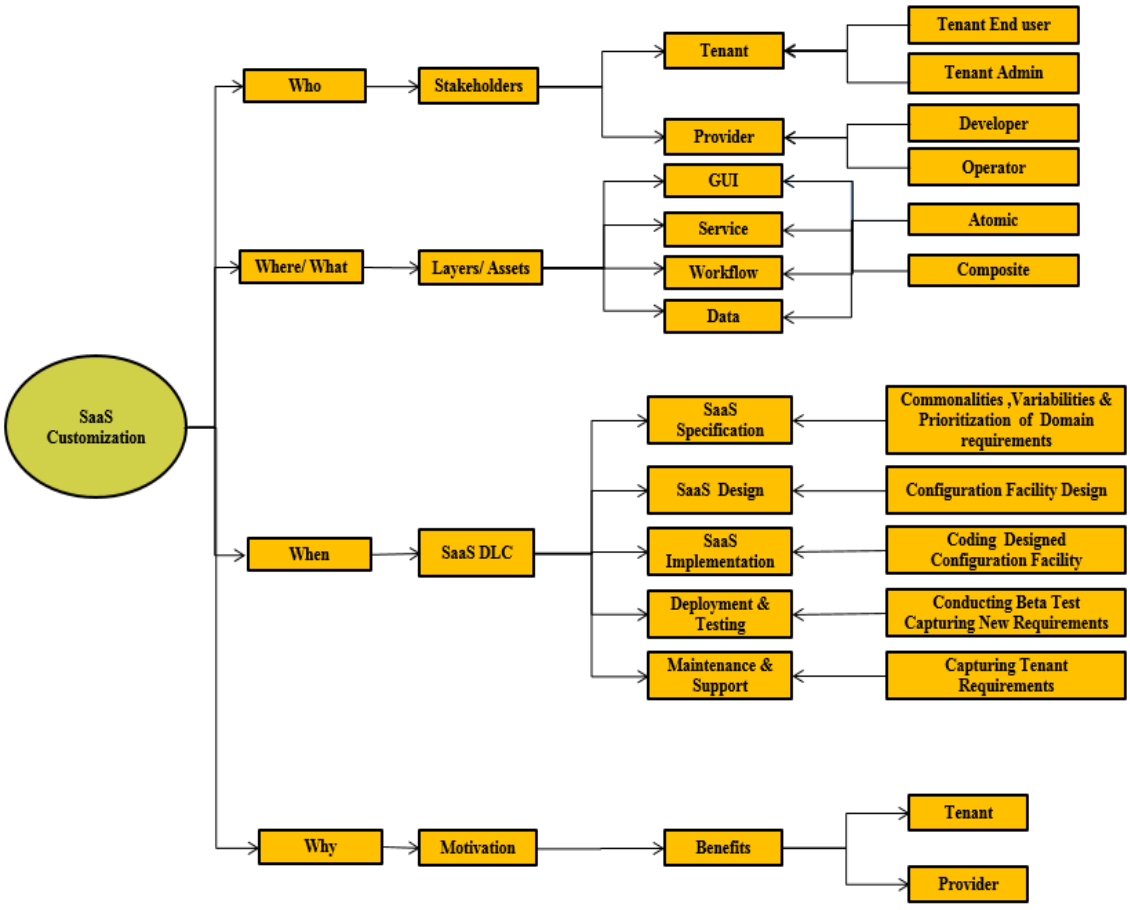

Figure 2. A Taxonomy of SaaS Customization Derived from 5 Ws

4.1 People Involved in SaaS Customization

Stakeholders who are involved in SaaS customization can be classified into the following categories:

4.1.1 SaaS Application Provider

It is the company responsible for the growth, management, and sale of SaaS application. This might be made more specific by terming the team responsible for the initial growth of the SaaS application the 'vendor,' but this is still part of the SaaS application provider's company [3].

The following stakeholders can be involved in SaaS customization and are employees of the SaaS provider [4]:

- **SaaS architect and developer** are responsible for designing and developing a multi-tenant SaaS application and are, in a way, the first to customize the SaaS application. Also, the SaaS developers should implement support for tenant-aware user management (e.g., to assign the tenant administrator role), and for managing the different features and configurations (i.e., configuration interface).
- **SaaS operator** is responsible for deploying and managing the SaaS application. The SaaS operator is also responsible for the maintenance of the SaaS application as well as the underpinning platform, and for keeping these up-to-date. The SaaS operator will have to verify the configurations of all tenants that may affect the complete architecture of the SaaS application and even conflict with the requirements of some tenants.

Maximum customizability is the most important element for SaaS application provider. Customizable elements can only be established once a domain's common requirements have been identified. An initial degree of customization could be carried out by the provider so that the tenant is not forced to start from scratch.

4.1.2 SaaS Application Tenant

It refers to the person seeking a SaaS application, to whom these stakeholders are essential:

- **Tenant end user** who usually does not have the appropriate training to do difficult customizations, and so may need to pay another team to complete such tasks. This would prevent the need for expensive training courses and mean that applications could be turned out quickly.
- **Tenant administrator** who/which is the person or company assigned to manage the SaaS application on behalf of the tenant (e.g., user management, access control, and configuration). These people have the relevant knowledge to customize applications, which the tenant user may lack. For instance, a study describes [5] the people or company specialized in understanding how existing SaaS applications work and how they can be adapted as consultants.

It is clear that the various requirements and levels of ability among the stakeholders mean that there is much to be taken into account when customizing SaaS application. These distinct theoretical categories may be more fluid in reality with, for example, the roles of the tenant provider and tenant administrator being taken by the same company or person. In some cases, they may also be the tenant user. However, for the sake of clarity, theoretically, this paper works as though they are distinct entities, as suggested above. Figure 3 outlines the further subcategories of these groups.

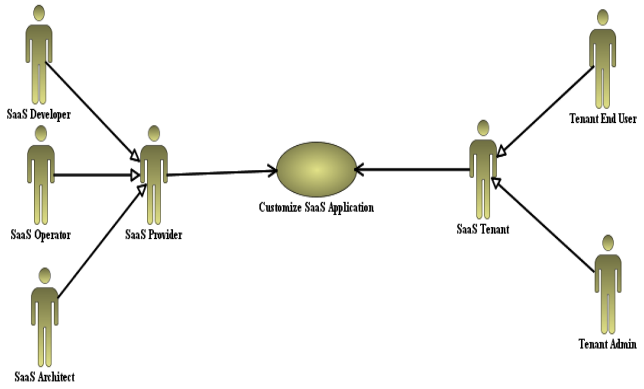

Figure 3. Overview of People Involved in SaaS Customization

4.2 Assets and Layers to be Customized

Even though What and Where questions are different regarding their formations, their answers are interlinked. Therefore, answering these questions would be combined together. The 'where' information refers to the SaaS application layer and the 'what' information refers to the SaaS application assets located in each layer.

Figure 4 outlines the structure of, customizable SaaS application. Every layer of the application must be customizable, from the Graphic User Interface (GUI) at the topic of the structure to the data at the bottom. All levels - functional and design-based - are impacted by customization.

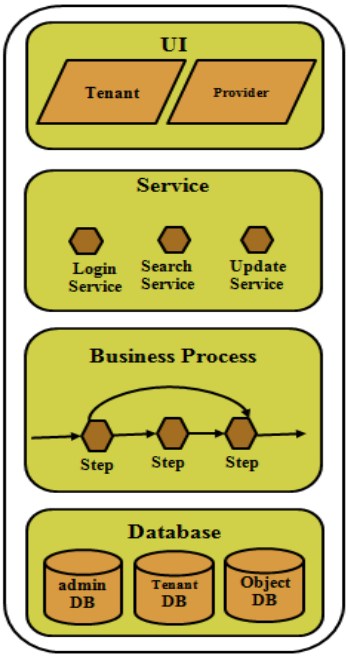

Figure 4. Customizable SaaS Layered Architecture

4.2.1 GUI Customization

The most obvious and basic customization concerns the GUI layer, which is responsible for communicating between systems and end users, may be modified by end users and provides the results of such modifications [5, 6]. The GUI's layout and theme may be customized [7, 8, 9]. It is possible to insert, remove and adjust elements on the forms or on the GUI itself [7, 8]. It is essential that each tenant in a multi-tenancy application can make their own changes while leaving those of others untouched [10].

The elements to be changed fall into one of the two following categories: atomic GUI components and composed GUI components [5]. These are outlined below:

- Atomic GUI components: password facilities, search tools, navigation aids, etc.
- Composed GUI components: functions made up of multiple atomic GUI components. This could enable a more comprehensive search tool, allowing the user to make various specifications (e.g., start and end dates).

4.2.2 Service Customization

This refers to both atomic and composite services [6]:

- Atomic service - entry-level, only completes basic commands.
- Composite service - multiple atomic services are combined to enable the completion of more difficult tasks.

4.2.3 Workflow Customization

The necessary assets to be customized fall into two categories:

- Atomic workflow: one which adheres to general business requirements [4], the templates of which may be created by the tenant based on existing services [4]. This means that workflow customization and service layers must be considered jointly [7, 11].
- Composed workflow: customization at the business logic level requires that the business logic code is adapted to the customer [12]. In addition, changing the workflow may change the application behavior [6, 9].

4.2.4 Data Customization

At the data level, a customizable application typically enables the creation of new attributes for existing entities or, less often, it may even allow for the creation of new entity types [10]. Data entities can be classified into atomic data entities and composed data entities [5]:

- Atomic data entities are concerned with the smallest details. The data is mostly numeric or string literals which lack internal structures (e.g., FirstName). They are restricted by specific datatype properties.
- Composed data entities comprise of data which is in some way related to atomic or other composed data (e.g., FirstName and LastName).

4.3 Customization within SaaS DLC

A conventional SLDC and SaaS application have similar development lifecycles. Some of the stages are assigned as different levels of importance or may take slightly longer to complete. There are also extra levels in SaaS development, including evaluation, subscribing and operations: these are less significant in, for example, an on-premise deployment. They are, however, essential for the development and management of SaaS application [13]. Accordingly, the customization must take place in some of SaaS DLC, either by customizing the SaaS application or by considering future customization, as shown in Figure 5.

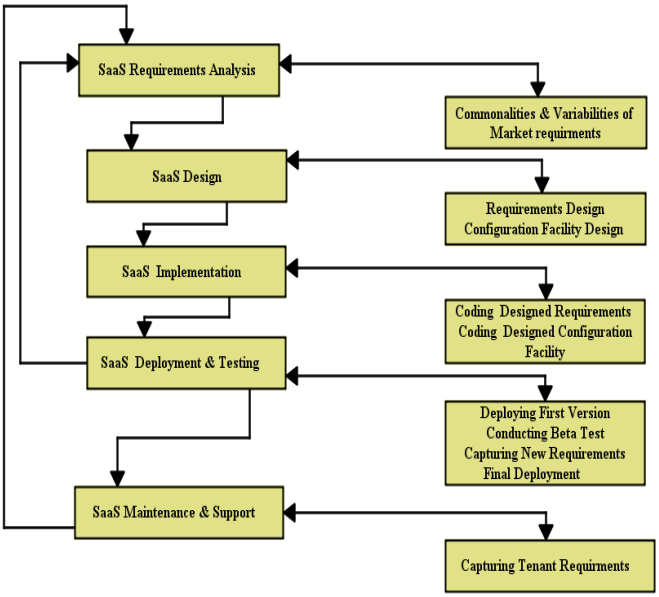

Figure 5. Overview of SaaS Customization within SaaS DLC

4.3.1 SaaS Specification

It is necessary to identify the software functions which will best address the customer's requirements. When distributed as a service, an application or platform must address the relevant business model, meaning that both the vendor's needs and the market constraints must be considered. A service application is often the result of a business adjusting to market constraints, as opposed to being for an individual [1]. The result of this stage is the eliciting of typical commonalities, as well as the variability of domain requirements. Requirement prioritization for customization has to be considered at this stage as well.

4.3.2 SaaS Design

Customer needs are constantly altering, and so SaaS application should be designed to be as adjustable as is necessary. The design of the configuration management facility has to be considered at this stage.

4.3.3 SaaS Implementation

This is concerned with coding the application. It is essential to be aware of the various levels of the application, including default, advanced and future settings. Moreover, it is necessary to consider coding the flexible design of the configuration management facility.

4.3.4 SaaS Deployment and Testing

After successful testing, the product is delivered or deployed to the tenant for their use. As soon as the SaaS application is given to the tenants, they will first do the beta testing. If any changes are required or if any bugs are caught, then an iteration of SaaS requirements analysis, design, and implementation begin to make these changes. Once those changes are made, or the bugs are fixed, then the final deployment will happen.

4.3.5 SaaS Maintenance & Support

It is essential that the service-level agreement (SLA) for maintaining SaaS application is as up-to-date with current conditions as possible. Therefore, tailoring the existing SaaS application to suit the newly introduced needs of a specific tenant falls under this stage. When the tenant has new requirements which are not yet supported by existing SaaS, a new development iteration of SaaS DLC might be started, potentially including requirement analysis, SaaS application design and implementation, and finally updates to the application deployment. The difference between this iteration and the previous one is that the results of the first iteration are implemented and deployed at the application level, whereas this one is implemented and deployed on a tenant-specific level.

4.4 Benefits and Motivations

The more customizable SaaS application we have, the greater its benefits we get. These are different depending on the stakeholder being considered. The following section of this paper considers the potential benefits of a customizable SaaS application for the SaaS application provider and the SaaS application tenant (end user and administrator).

4.4.1 SaaS Provider Benefits

SaaS providers will realize several benefits from developing a highly customizable SaaS application, some of them are outlined below:

1. It will be possible for the SaaS application to address the various needs of the SaaS tenant, which may change or be unclear at the outset.
2. It will lengthen the period of use of a software system. If an application can be altered as the user's requirements change, there is no need for it to be replaced.
3. It will meet the needs of a larger portion of the market by making potential users aware of the sheer number of customizable elements.
4. It may require less maintenance, and those tasks left may be delegated to SaaS tenant.
5. It will expand the available enhancements by a customization mechanism. This will be more cost-effective, as it will require less time on the part of the SaaS developer.

4.4.2 SaaS Tenant Benefits

The SaaS tenant will be on the receiving end of the following benefits:

1. There will be a renewed distinctiveness of the SaaS application, and the tenant will feel in control, able to make changes to the functionality as necessary.
2. The tenant will obtain a new understanding of the application and its domain.
3. A user community may result from the greater ease of communication between users and developers created by the software's customizability. This would be a space to seek help and try out new ideas.
4. A customization mechanism could be created to reduce the amount of money spent on IT maintenance and development.

5. Conclusion

This paper has addressed the core elements of SaaS customization as illustrated by the Five Ws approach, which is summarized in Figure 1. By answering these questions and organizing the results, the taxonomy of SaaS customization is structured as illustrated in Figure 2. This taxonomy shows the main players in SaaS customization, assets and layers of SaaS application to be customized, and the benefits of a highly customizable SaaS application. Furthermore, it shows when customization must take place in SaaS DLC, either by customizing the SaaS application or by considering future customization.

This study can guide the SaaS implementation team throughout SaaS DLC when customization is inescapable. This study is limited to the examination of SaaS customization in term of reporting on the Five Ws of SaaS customization derived from previous research.

In conclusion, of course, there are still some challenges to a highly customizable SaaS application, which will always stall researchers and service providers. These are considered in our future work. At the same time, future studies may also want to examine the SaaS customization by addressing the question of 'How?' and then to associate it with this study.

References

[1] C. Lizhen, W. Haiyang, J. Lin, H. Pu, “Customization modeling based on metagraph for multi-tenant applications”, in *2010 5th International Conference on Pervasive Computing and Applications (ICPCA)*, IEEE, pp. 255–260, 2010.

[2] K. Zhang, X. Zhang, S. Wei, H. Liang, Y. Huang, Z. Liangzhao, L. Xuanzhe, “A policy-driven approach for software-as-services customization”, in *The 9th IEEE International Conference on E-Commerce Technology and The 4th IEEE International Conference on Enterprise Computing, E-Commerce and E-Services (CEC- EEE 2007)*. IEEE, pp. 123–130, 2007.

[3] M. M. Al-Shardan, D. Ziani, “Configuration as a service in multi-tenant enterprise resource planning system”, *Lecture Notes on Software Engineering*, vol. 3, no. 2, p. 95, 2015.

[4] S. Walraven, D. Van Landuyt, E. Truyen, K. Handekyn, W. Joosen, "Efficient customization of multi-tenant software-as-a-service applications with service lines", *Journal of Systems and Software*, vol. 91, pp. 48–62, 2014.

[5] W.-T. Tsai, X. Sun, "SaaS multi-tenant application customization", in *Service-Oriented System Engineering (SOSE), 2013 IEEE 7th International Symposium on*. IEEE, pp. 1–12, 2013.

[6] W.-T. Tsai, Q. Shao, W. Li, "Oic: Ontology-based intelligent customization framework for SaaS", in *2010 IEEE International Conference on Service-Oriented Computing and Applications (SOCA)*. IEEE, pp. 1–8, 2010.

[7] M. Mcsi, "Configurability in SaaS (software as a service) applications", in *Proc. 2nd Annual Conference on India Software Engineering Conference*, pp. 19–26, 2009.

[8] P. K. Arya, V. Venkatesakumar, S. Palaniswami, "Configurability in SaaS for an electronic contract management application", in *Proceedings of the 12th international conference on Networking, VLSI and signal processing*, World Scientific and Engineering Academy and Society (WSEAS), pp. 210–216, 2010.

[9] K. Czarnecki, S. Helsen, U. Eisenecker, "Staged configuration through specialization and multilevel configuration of feature models", *Software Process: Improvement and Practice*, vol. 10, no. 2, pp. 143–169, 2005.

[10] A. Correia, J. R. Penha, A. M. R. da Cruz, "An architectural model for customizing the business logic of SaaS applications." in *ICSOFT*, pp. 162– 168, 2013.

[11] S. Kang, S. Kang, S. Hur, "A design of the conceptual architecture for a multi-tenant SaaS application platform", in *Computers, Networks, Systems and Industrial Engineering (CNSI), 2011 First ACIS/JNU International Conference on*. IEEE, pp. 462–467,2011.

[12] A.Q. Ali, A.B.M. Sultan, A.A.A. Ghani, H. Zulzalil, "Critical issues across SaaS development: learning from experience", *International Journal of Advances in Electronics and Computer Science*, vol. 4, no. 9, pp. 69–74, 2017.

[13] G. Kulkarni, P. Chavan, H. Bankar, K. Koli, V. Waykule, "A new approach to software as a service cloud", in *2012 7th International Conference on Telecommunication Systems, Services, and Applications (TSSA)*, pp. 196–199, 2012.