Supplement file for the Manuscript titled ”An empirical investigation of software customization and its impact on the quality of Software as a Service: perspectives from software professionals”, submitted for Applied Sciences (ISSN 2076-3417)

Appendix A. Measurement items

Construct	Item		Reference*
Configuration	Con 1	Configuration typically maintains diversity by establishing predefined parameters, options, and components, and treats each tenant individually.	[1–5]
	Con 2	Each tenant can configure the application in a standalone way by employing techniques to modify the functions of applications within established limits.	[1, 6, 7]
	Con 3	SaaS providers have to develop and capture sets of services and plugins, from which tenants can make selections and perform configurations.	[8, 9]
	Con 4	Tenants can create customization based on templates.	[10–16]
	Con 5	Tenants can select their desired workflow templates and items relating to SaaS application templates from the template repository.	[11, 12, 16]
	Con 6	When a tenant wishes to subscribe to the SaaS application, the capabilities of each feature within the system are analyzed to determine whether they ought to be assimilated within the application.	[9, 17]
	Con 7	All Configurations established by the tenants have to be within the context of the runtime of the application.	[1, 18–20]
	Con 8	An option of disabling or excluding some features of the SaaS application should be provided with the isolation effect across the tenants.	[21, 22]
Composition	Com 1	The multiple interacting components of the SaaS application are consolidated, and new application components can be shared between multiple SaaS tenants and end users.	[12, 22–27]
	Com 2	Composing different collaboration components is done according to the runtime of the SaaS application.	[22, 23, 28–30]
	Com 3	The composition of components takes into account the subcomponents of the core one.	[28, 31, 32]
	Com 4	Performing the composition of SaaS application components considers the relationships and dependencies between these components.	[4, 7, 23]
Extension	Ext 1	The SaaS application is extended by adding custom code to extend the application through custom functionality.	[5, 12, 33]
	Ext 2	The SaaS application provides a set of extension points which permit a customized service to be plugged in at virtual points in the application.	[5, 33]
	Ext 3	Injecting custom code into SaaS application has to be supported at the run time of the application.	[19, 34]
	Ext 4	The SaaS service provider supplies an open platform and an API, which allows developers to inject custom codes into business object layers.	[8, 35]
	Ext 5	These injected codes can either be replacements for existing objects or extensions to them.	[35]

Continued on next page...

Construct	Item		Reference*
	Ext 6	An extension may be private to an individual tenant or shared by multiple tenants.	[36]
Integration	Int 1	SaaS application functionality can be expanded through the addition of extra services via external SaaS providers.	[9, 37–40]
	Int 2	SaaS service customers assume that the SaaS application will be easy to amalgamate with their existing in-house systems	[35, 37, 38, 40, 41]
	Int 3	Integration encompasses aspects which ensure a smooth flow at both design time and runtime.	[37, 38]
	Int 4	Integration platforms incorporate both service framework, through which services can be assimilated, and process framework, through which business processes can be executed.	[37, 38]
	Int 5	Additional services from third-party SaaS providers employ different programming languages running in different environments.	[40]
	Int 6	Coding or scripting is utilized to incorporate external services into SaaS application.	[37]
	Int 7	Incorporating services into SaaS application requires an integration interface in the form of conguration or setup.	[37, 38]
	Int 8	Synchronization toolkits and data retrieval mechanisms are created to respond to the demands posed by integration.	[38, 41]
Modification	Mod 1	Source code modifications are made to SaaS application to generate a new functionality without changing a shared code base.	[19, 42, 43]
	Mod 2	The code modification must take resources allocation for customized code into account, ensuring operational cost-efficiency in terms of maintenance costs and resource sharing among tenants.	[19, 42]
	Mod 3	SaaS vendors must manage all elements of customization codes on an individual tenant basis without developing many software versions for each tenant.	[42]
	Mod 4	SaaS vendors alter application codes when identical customizations are defined and justified by a considerable number of tenants.	[19, 42]
	Mod 5	Source code modifications are made by adding/deleting methods or attributes, or by changing the objects current implementation methods.	[44, 45]
SaaS Quality	QA 1	**Multi-tenancy:** SaaS services can support instances of simultaneous access by multiple users for multiple tenants.	[46, 47]
	QA 2	**Scalability:** SaaS providers can manage growth or decline in the level of services.	[46, 48–52]
	QA 3	**Availability:** SaaS services can function within a specific time to satisfy users needs.	[46, 48, 51–54]
	QA 4	**Reliability:** SaaS application maintains operating and functioning under given conditions without failure within a given time period.	[46, 48, 49, 51–54]
	QA 5	**Maintainability:** Modifications to the application are made by SaaS provider to retain it in the condition of good repair.	[46, 52, 53]
	QA 6	**Security:** The effectiveness of SaaS provider's controls on service data, access to the services, and the physical facilities from which service are provided.	[46, 52]
	QA 7	**Usability:** The ease with which SaaS application can be used to achieve tenant-specific-goal.	[46, 54]

Continued on next page...

Construct	Item		Reference*
	QA 8	**Interoperability:** SaaS service can easily interact with other services from the same SaaS provider or other providers.	[46, 52, 53]
	QA 9	**Efficiency:** SaaS services effectively utilize resources to perform their functions.	[46, 48, 49, 51]
	QA 10	**Functionality:** SaaS application provides an extensive set of features.	[46, 52]
	QA 11	**Accessibility:** SaaS services are operable by users with different disabilities.	[46, 52, 53]
	QA 12	**Commonality:** SaaS services possess common features and are amenable to reuse by multiple users.	[46–49]
	QA 13	**Response time:** SaaS application adheres to a defined time limit between service request and service response.	[46, 52, 55–59]

* All measurement items used in this study were drawn on the findings of our previously-conducted studies [60, 61], however, we reported the references that represent the main sources for each item.

Appendix B: Rounds of measurement model modification through CFA

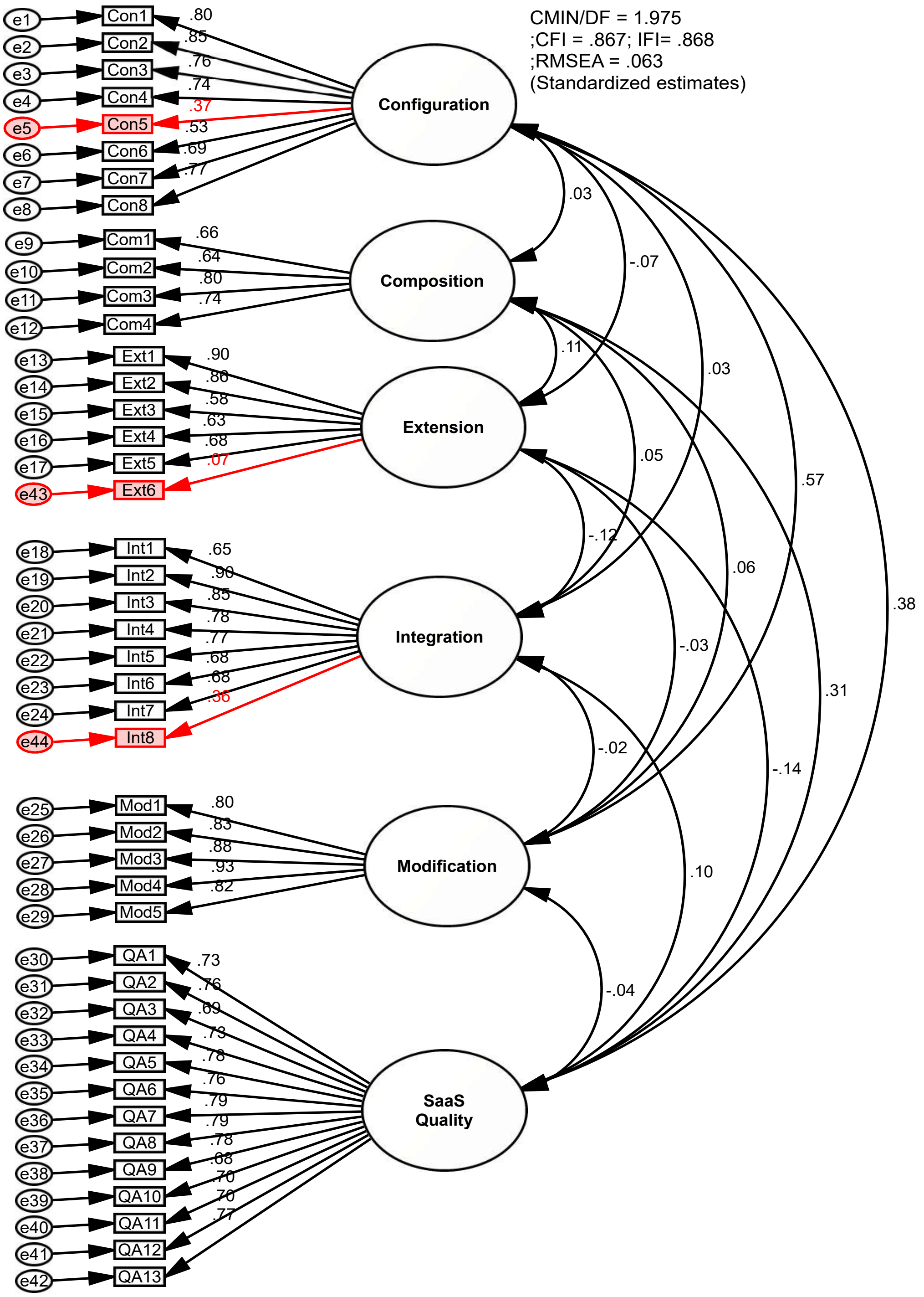

Figure 1: The initial measurement model combining all six constructs

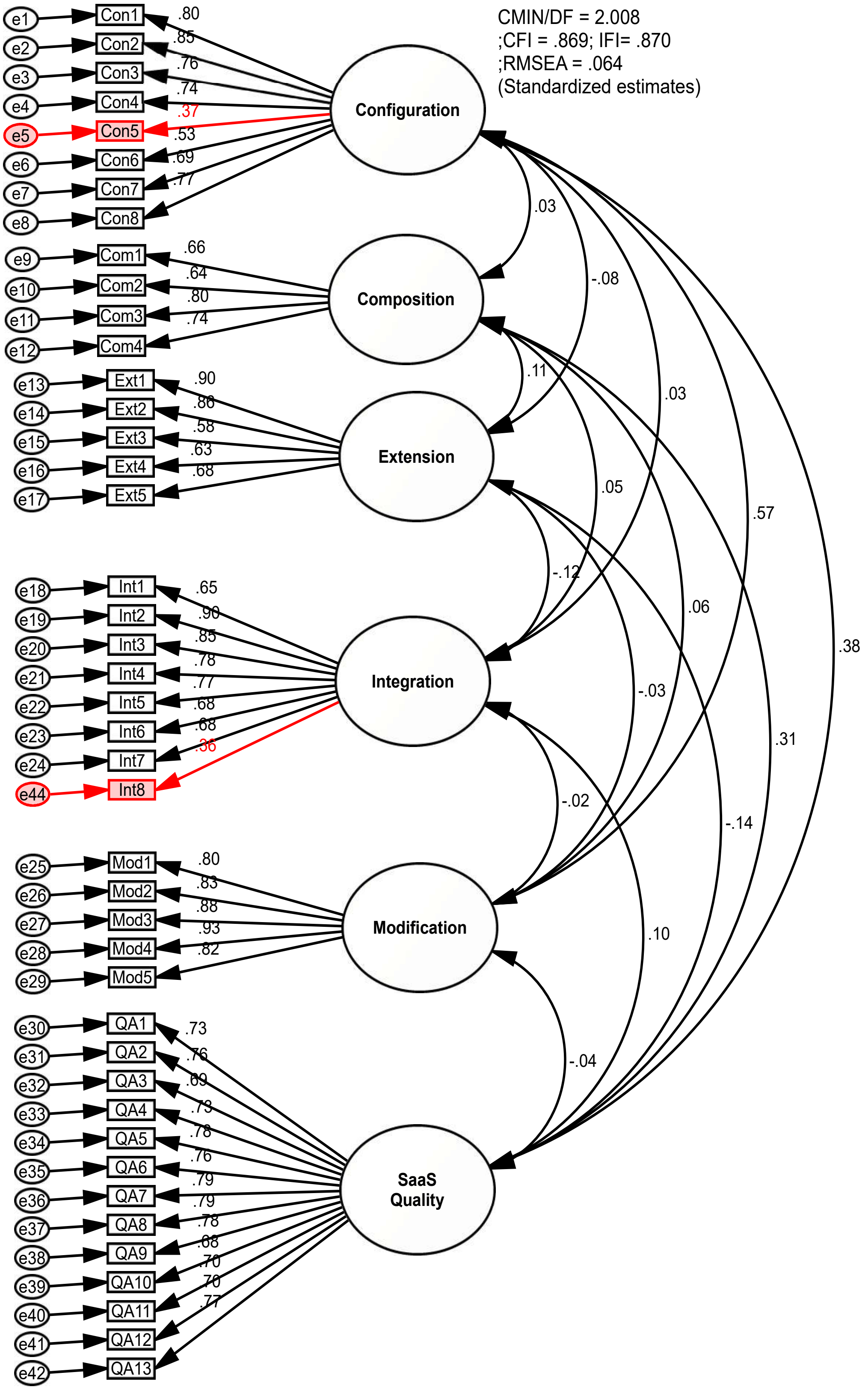

Figure 2: The measurement model combining all six constructs (Version 1)

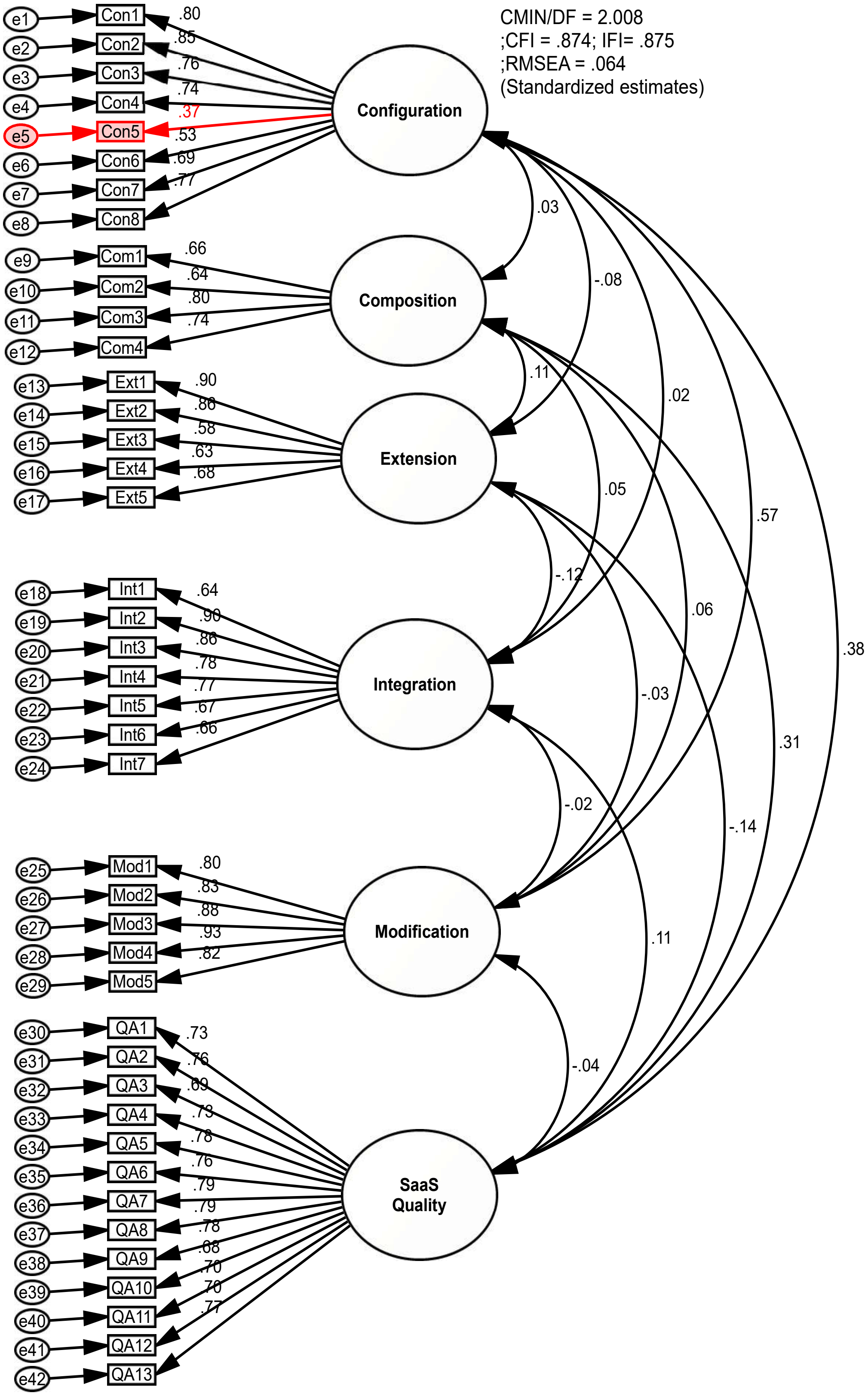

Figure 3: The measurement model combining all six constructs (Version 2)

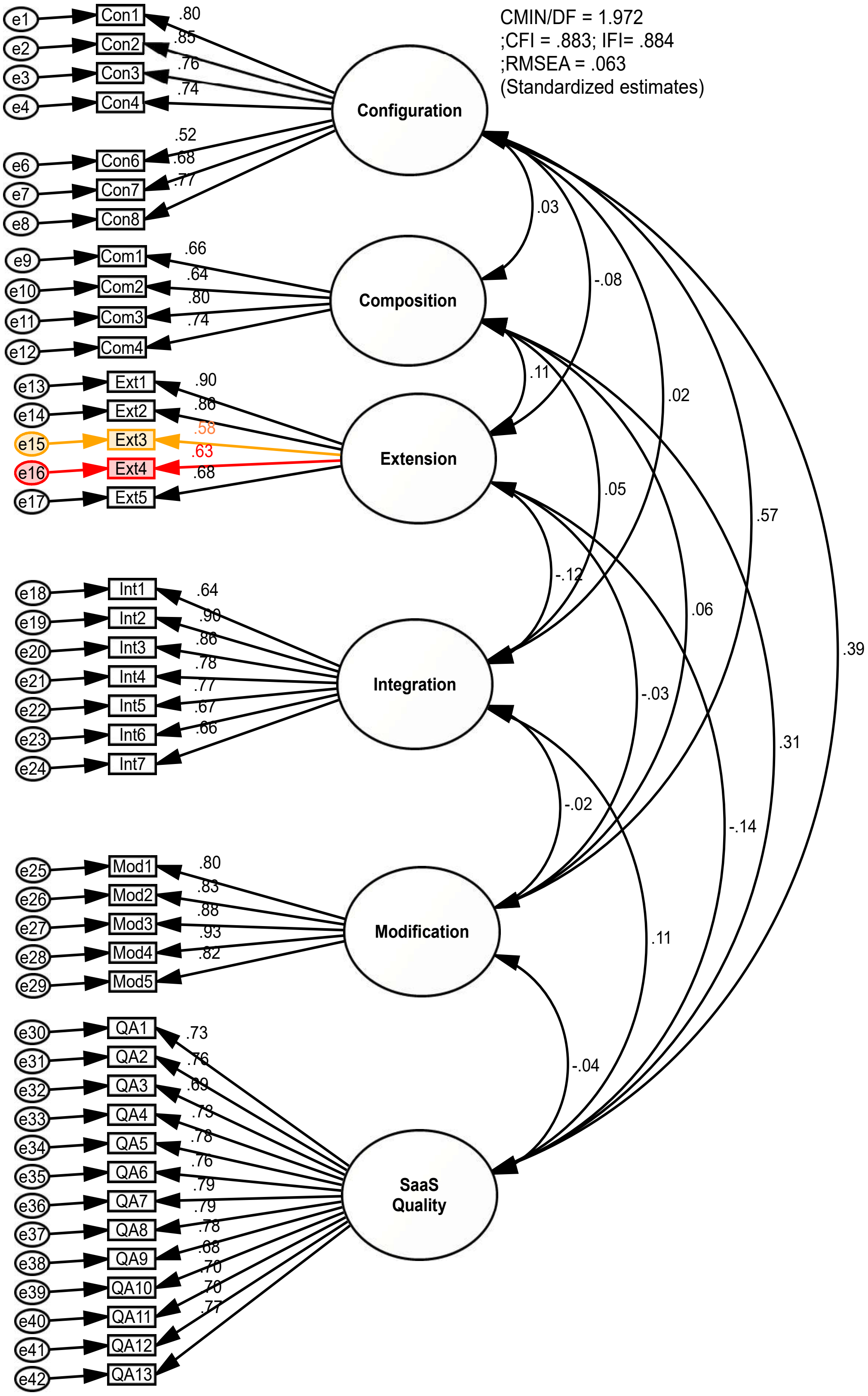

Figure 4: The measurement model combining all six constructs (Version 3)

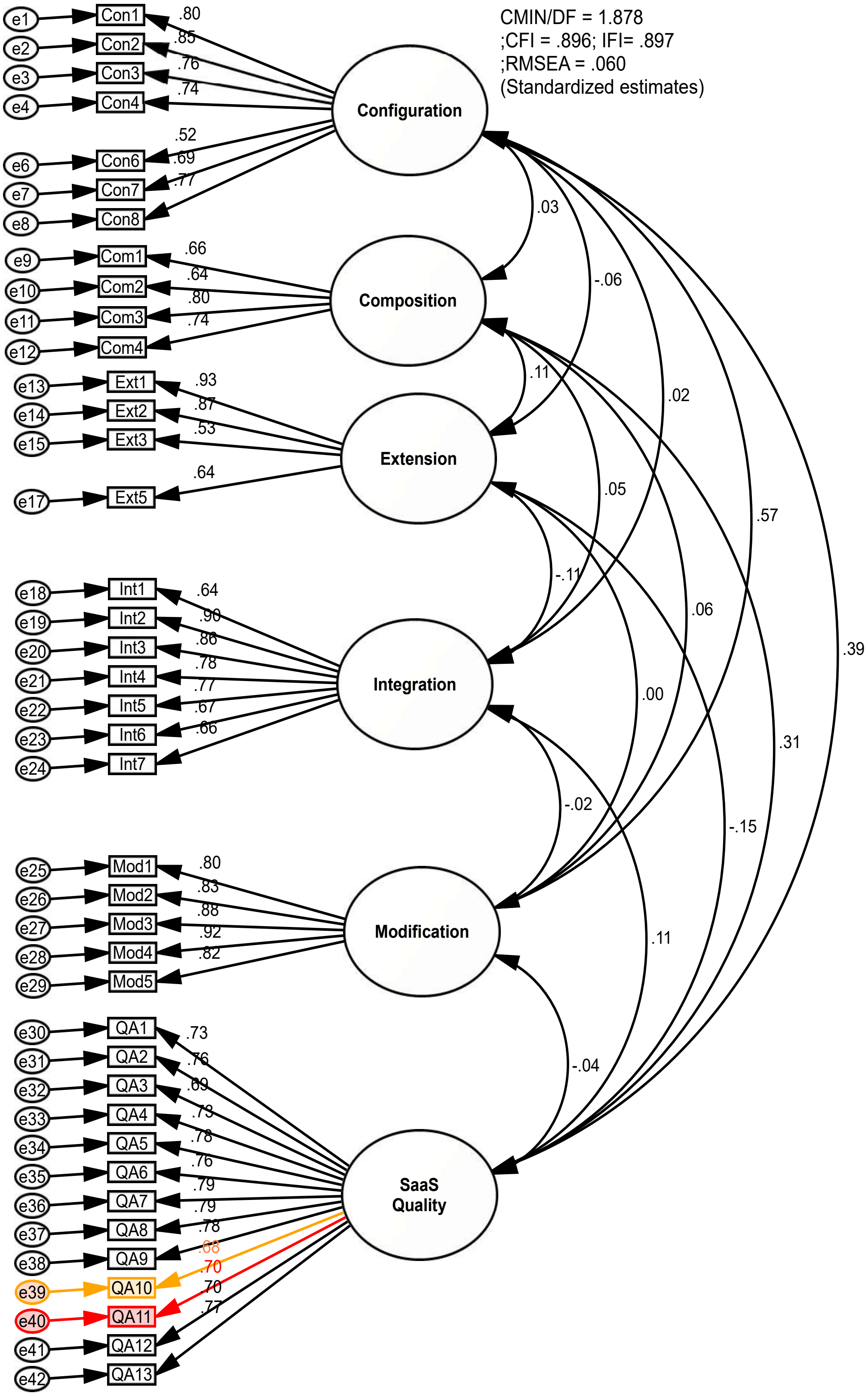

Figure 5: The measurement model combining all six constructs (Version 4)

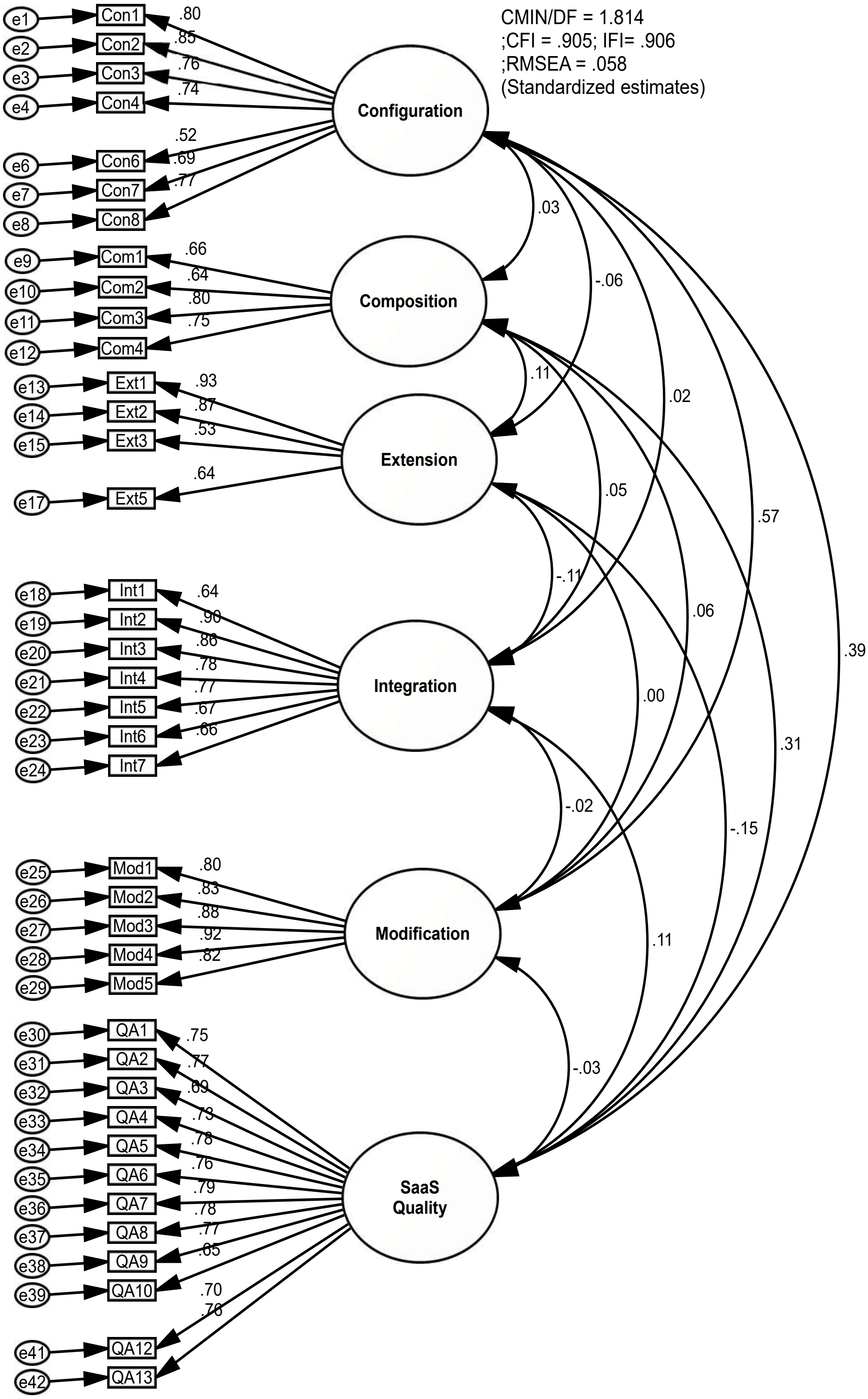

Figure 6: The final measurement model combining all six constructs (Version 5)

Appendix C: Structural model

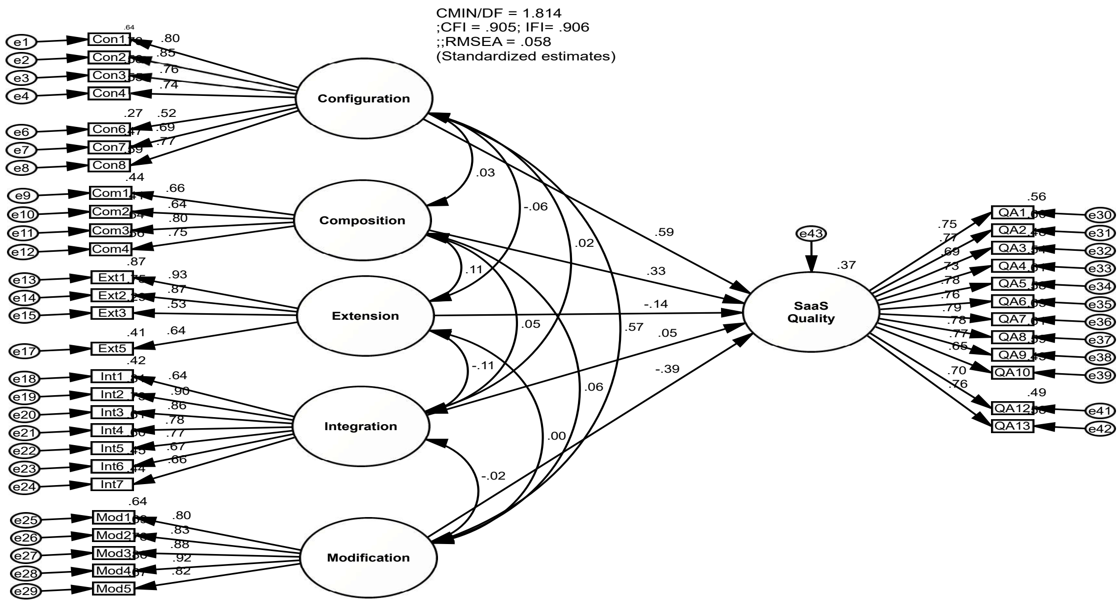

Figure 7: The intial Structural model

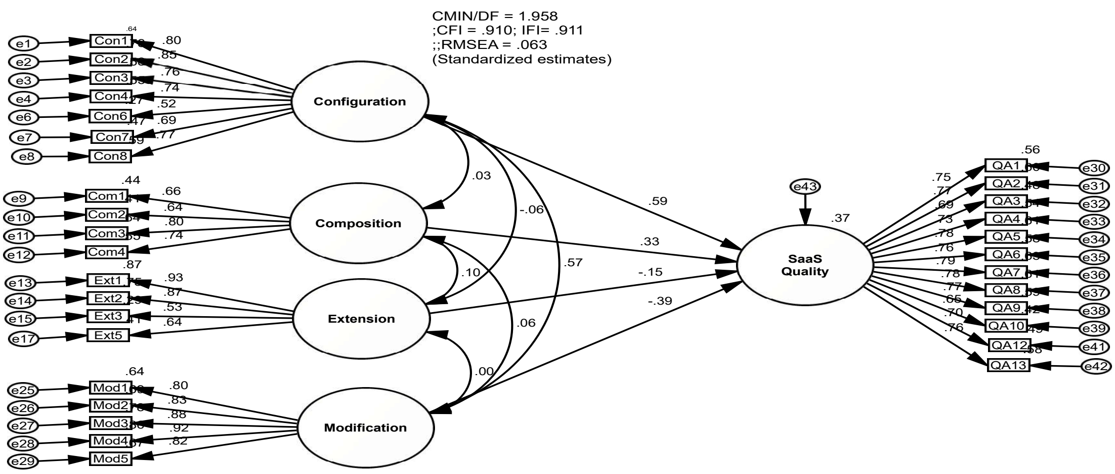

Figure 8: Structural model without integration construct

Appendix D: Exploratory Factor Analysis

Exploratory factor analysis (EFA) was performed in order to validate the structure of the model via factor extraction and rotation , however, some tests should be considered before. The first step in conducting an EFA assessment is to determine the appropriateness of the sample size. In this study, the Kaiser-Meyer-Olkin Measure of Sampling Adequacy (KMO) and Bartlett's Test were used for this purpose. SPSS was used to perform EFA, as well as to perform Principal Components Analysis (PCA) and Varimax rotation.

1. **KMO and Bartlett's Test of Sphericity**

   Table 2 lists the results of KMO and Bartletts Test. KMO, used for the measurement of sample adequacy, yielded a value of 0.865, and Bartlett's test of sphericity yielded a result of p <0.001, indicating that the amount of data for analysis is acceptable.

   Table 2: KMO and Bartlett's Test of Sphericity statistics

KMO Measure of Sampling Adequacy		0.865
Bartlett's Test of Sphericity	Approx. Chi-Square	6845.875
	df	861.000
	Sig.	0.000

2. **Communality**

   Communality is another aspect of factor analysis specifically for common factor analysis. To conduct PCA, the communality (common variance) is used to aggregate indicators to produce constructs. In this study, the communalities of items varied from 0.044 (Ext6) to 0.845 (Mod4). For an item remain in the model, the communality should be greater than 0.4. Low communality shows the factor model does not work well for that item and marks it as a candidate for removal from the study. It is noteworthy that all but two of the items in our model obtained a communality value above 0.4; the two outliers were item Ext6 (0.044) and item Int8 (0.276). These two items were removed from the model and the EFA was rerun, resulting in a new item Communality range from 0.472 (Con5) to 0.845 (Mod4), as shown in Table 3.

3. **Factor Extraction**

   To extract factors, principal component analysis was performed. The eigenvalue indicates the variance of a given factor in all variables. Using eigenvalues greater than 1.5, the analysis extracted six factors (Table 4), explaining 64.013 % of the total variance. The eigenvalues showed that the first factor (SaaS Quality) explained 19.222% of the variance, the second factor (Integration) 10.827% of the variance, the third factor (Modification) 9.993% of the variance and the fourth factor (Configuration) 9.930%. The fifth and sixth factors (Extension and Composition) had eigenvalues of over one, each factor explaining about 7.929% and 6.130% of the variance, respectively.

4. **Factor Rotation**

   A rotational strategy was done to identify the factors necessary for a clear pattern of loading. Varimax rotation was selected to maximize the variance on the new axes. Using the Rotated Component Matrix, the factor was extracted. Table 5 gives the results of this study, from which six factors, constructs of this study, were extracted. It can be seen that all items attracted coefficients of more than 0.4, therefore all were retained for further analysis. The EFA results showed that all items have high loadings on their hypothesized constructs, and confirm the six constructs of the study as follows: 13 items loaded onto the first factor (SaaS Quality), 7 items loaded onto the second factor (Integration), 5 items loaded onto the third factor (Modification), 8 items loaded onto the fourth factor (Configuration), 5 items loaded onto the fifth factor (Extension), and 4 items loaded onto the sixth factor (Composition).

Table 3: Communalities statistics

Items	Communality	Items	Communality
Con1	0.627	Int5	0.655
Con2	0.721	Int6	0.605
Con3	0.619	Int7	0.612
Con4	0.569	Int8*	0.276
Con5	0.473	Mod1	0.712
Con6	0.591	Mod2	0.771
Con7	0.541	Mod3	0.813
Con8	0.647	Mod4	0.845
Com1	0.598	Mod5	0.742
Com2	0.566	QA1	0.594
Com3	0.704	QA2	0.617
Com4	0.661	QA3	0.538
Ext1	0.743	QA4	0.569
Ext2	0.678	QA5	0.641
Ext3	0.563	QA6	0.607
Ext4	0.656	QA7	0.644
Ext5	0.646	QA8	0.650
Ext6*	0.044	QA9	0.637
Int1	0.565	QA10	0.569
Int2	0.753	QA11	0.592
Int3	0.681	QA12	0.532
Int4	0.658	QA13	0.623

* Ext6 and Int8 are also dropped from the measurement model through CFA due to they had the factor loading of 0.07 and 0.36 respectively which were the lowest factor loadings in the model and less than the acceptable item loading value (>0.5).

Table 4: Total number of factor extracted in EFA

Component	**Initial Eigenvalues**			**Extraction Sums of Squared Loadings**			**Rotation Sums of Squared Loadings**		
	Total	**% of Variance**	**Cumulative %**	**Total**	**% of Variance**	**Cumulative %**	**Total**	**% of Variance**	**Cumulative %**
1	9.159	21.808	21.808	9.159	21.808	21.808	8.073	19.222	19.222
2	5.776	13.751	35.559	5.776	13.751	35.559	4.547	10.827	30.049
3	4.549	10.831	46.390	4.549	10.831	46.390	4.197	9.993	40.042
4	3.368	8.019	54.409	3.368	8.019	54.409	4.170	9.930	49.972
5	2.339	5.568	59.977	2.339	5.568	59.977	3.330	7.929	57.901
6	1.703	4.054	64.031	1.703	4.054	64.031	2.575	6.130	64.031

Extraction Method: Principal Component Analysis.

Table 5: Rotated component matrix

Items	Component 1	2	3	4	5	6
QA8	0.798					
QA9	0.788					
QA5	0.788					
QA7	0.786					
QA13	0.778					
QA6	0.769					
QA11	0.762					
QA2	0.756					
QA4	0.734					
QA1	0.732					
QA10	0.727					
QA12	0.703					
QA3	0.698					
Int2		0.872				
Int4		0.824				
Int3		0.821				
Int5		0.814				
Int6		0.768				
Int7		0.755				
Int1		0.730				
Mod4			0.876			
Mod3			0.869			
Mod2			0.834			
Mod5			0.829			
Mod1			0.807			
Con6				0.761		
Con2				0.735		
Con8				0.711		
Con1				0.678		
Con7				0.657		
Con5				0.630		
Con3				0.624		
Con4				0.609		
Ext1					0.849	
Ext2					0.813	
Ext5					0.799	
Ext4					0.796	
Ext3					0.726	
Com3						0.820
Com4						0.759
Com1						0.759
Com2						0.732

Extraction Method: Principal Component Analysis.
Rotation Method: Varimax with Kaiser Normalization
Rotation converged in 6 iterations.

Appendix E: Common latent factor (CLF) procedure

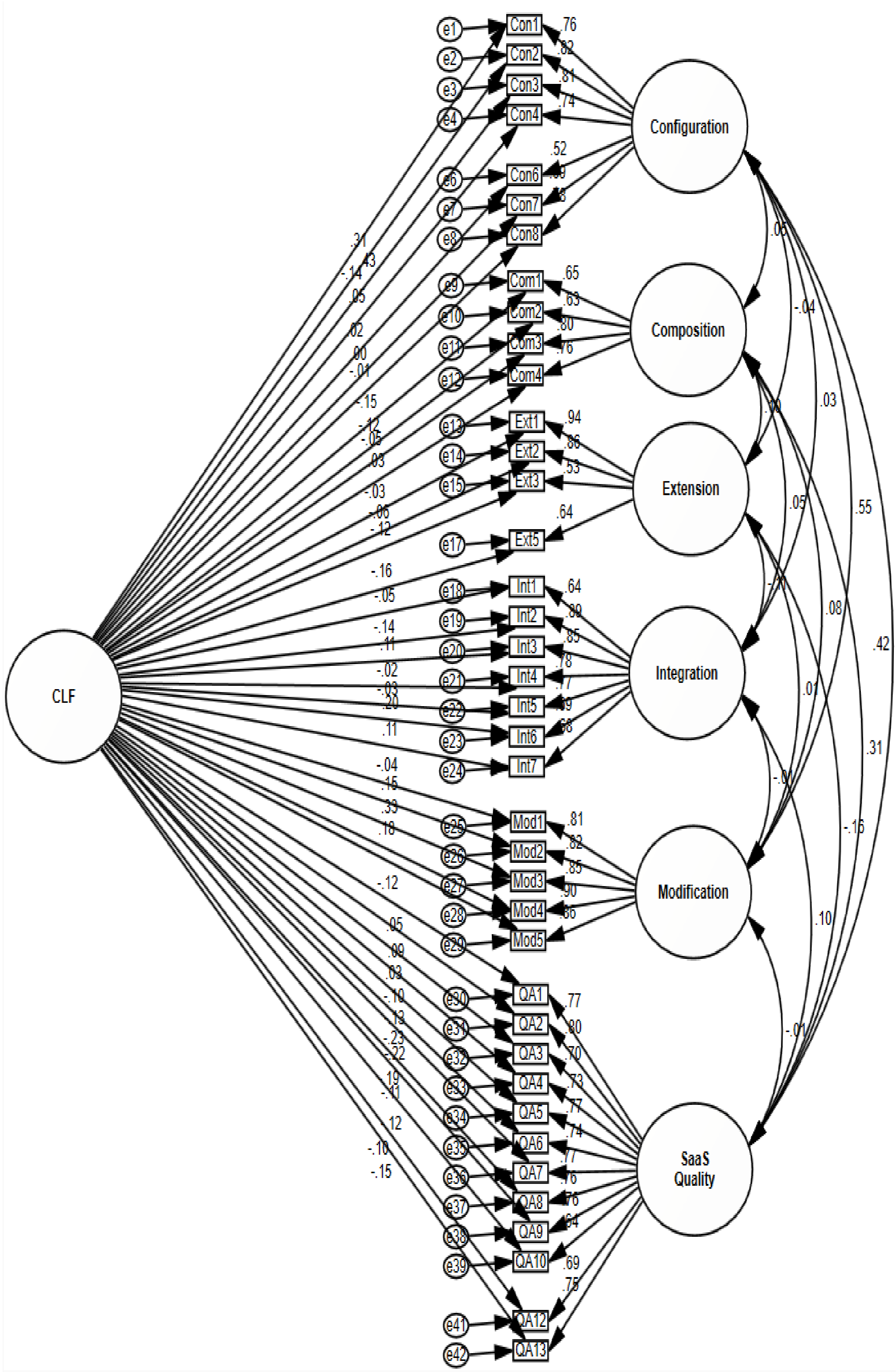

Figure 9: Measurement Model with CLF

Table 6: The measurement model with and without CLF

Items		**Constructs**	**Estimate with CLF**	**Estimate without CLF**	**Difference**
Con1	$<---$	Configuration	0.757	0.798	0.041
Con2	$<---$	Configuration	0.82	0.855	0.035
Con3	$<---$	Configuration	0.812	0.763	-0.049
Con4	$<---$	Configuration	0.739	0.74	0.001
Con6	$<---$	Configuration	0.52	0.519	-0.001
Con7	$<---$	Configuration	0.689	0.685	-0.004
Con8	$<---$	Configuration	0.776	0.767	-0.009
Com1	$<---$	Composition	0.65	0.66	0.01
Com2	$<---$	Composition	0.631	0.642	0.011
Com3	$<---$	Composition	0.795	0.8	0.005
Com4	$<---$	Composition	0.759	0.745	-0.014
Ext1	$<---$	Extension	0.935	0.932	-0.003
Ext2	$<---$	Extension	0.863	0.868	0.005
Ext3	$<---$	Extension	0.527	0.534	0.007
Ext5	$<---$	Extension	0.636	0.644	0.008
Int2	$<---$	Integration	0.895	0.901	0.006
Int3	$<---$	Integration	0.849	0.856	0.007
Int4	$<---$	Integration	0.781	0.782	0.001
Int5	$<---$	Integration	0.771	0.772	0.001
Int6	$<---$	Integration	0.691	0.669	-0.022
Int7	$<---$	Integration	0.679	0.665	-0.014
Int1	$<---$	Integration	0.644	0.644	0
Mod1	$<---$	Modification	0.814	0.799	-0.015
Mod2	$<---$	Modification	0.816	0.829	0.013
Mod3	$<---$	Modification	0.851	0.884	0.033
Mod4	$<---$	Modification	0.904	0.925	0.021
Mod5	$<---$	Modification	0.862	0.819	-0.043
QA2	$<---$	SaaS_Quality	0.799	0.774	-0.025
QA3	$<---$	SaaS_Quality	0.704	0.691	-0.013
QA4	$<---$	SaaS_Quality	0.727	0.734	0.007
QA5	$<---$	SaaS_Quality	0.77	0.78	0.01
QA6	$<---$	SaaS_Quality	0.738	0.761	0.023
QA7	$<---$	SaaS_Quality	0.773	0.795	0.022
QA8	$<---$	SaaS_Quality	0.763	0.783	0.02
QA1	$<---$	SaaS_Quality	0.765	0.747	-0.018
QA9	$<---$	SaaS_Quality	0.76	0.771	0.011
QA10	$<---$	SaaS_Quality	0.641	0.652	0.011
QA12	$<---$	SaaS_Quality	0.688	0.697	0.009
QA13	$<---$	SaaS_Quality	0.75	0.763	0.013

References

[1] M. Xin, N. Levina, Software-as-a-service model: Elaborating client-side adoption factors, in: Proceedings of the 29th International Conference on Information Systems, R. Boland, M. Limayem, B. Pentland,(eds), Paris, France, Paris, France, 2008. doi:https://dx.doi.org/10.2139/ssrn.1319488.
URL https://ssrn.com/abstract=1319488

[2] N. K. Salih, T. Zang, Modeling and self-configuring saas application, CoRR abs/1606.05991.

[3] J. Kabbedijk, S. Jansen, Variability in multi-tenant environments: Architectural design patterns from industry, in: Proceedings of the 30th International Conference on Advances in Conceptual Modeling: Recent Developments and New Directions, ER'11, Springer-Verlag, Berlin, Heidelberg, 2011, pp. 151–160.
URL http://dl.acm.org/citation.cfm?id=2075202.2075227

[4] A. A. Shahin, Multi-dimensional customization modelling based on metagraph for saas multi-tenant applications, CoRR abs/1402.6045. arXiv:1402.6045.
URL http://arxiv.org/abs/1402.6045

[5] R. Mietzner, F. Leymann, Generation of bpel customization processes for saas applications from variability descriptors, in: 2008 IEEE International Conference on Services Computing, Vol. 2, 2008, pp. 359–366. doi:10.1109/SCC.2008.85.

[6] K. ZHANG, X. ZHANG, W. SUN, H. LIANG, Y. HUANG, L. ZENG, X. LIU, 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), 2007, pp. 123–130. doi:10.1109/CEC-EEE.2007.9.

[7] H. Li, Y. Shi, Q. Li, A multi-granularity customization relationship model for saas, in: 2009 International Conference on Web Information Systems and Mining, 2009, pp. 611–615. doi:10.1109/WISM.2009.128.

[8] S. Zhao, Y. Zhang, B. Shen, X. Shen, R. Chen, Mass data processing and personalized services in shanghai e-commerce credit evaluation platform, in: 2014 IEEE International Conference on Progress in Informatics and Computing, 2014, pp. 481–485. doi:10.1109/PIC.2014.6972382.

[9] F. Mohamed, M. Abu-Matar, R. Mizouni, M. Al-Qutayri, Z. A. Mahmoud, Saas dynamic evolution based on model-driven software product lines, in: 2014 IEEE 6th International Conference on Cloud Computing Technology and Science, 2014, pp. 292–299. doi:10.1109/CloudCom.2014.131.

[10] W. 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), 2010, pp. 1–8. doi:10.1109/SOCA.2010.5707139.

[11] W. Tsai, Y. Huang, Q. Shao, Easysaas: A saas development framework, in: 2011 IEEE International Conference on Service-Oriented Computing and Applications (SOCA), 2011, pp. 1–4. doi:10.1109/SOCA.2011.6166262.

[12] A. I. Saleh, M. A. Fouad, M. Abu-Elkheir, Classifying requirements for variability optimization in multitenant applications, in: 2014 IEEE 6th International Conference on Cloud Computing Technology and Science, 2014, pp. 32–37. doi:10.1109/CloudCom.2014.142.

[13] M. Ralph, Using variability descriptors to describe customizable saas application templates, Institute of Architecture of Application Systems (2008) 1–27.

[14] D. Chen, Q. Li, L. Kong, Process customization framework in saas applications, in: 2013 10th Web Information System and Application Conference, 2013, pp. 471–474. doi:10.1109/WISA.2013.94.

[15] S. T. Ruehl, U. Andelfinger, Applying software product lines to create customizable software-as-a-service applications, in: Proceedings of the 15th International Software Product Line Conference, Volume 2, SPLC '11, ACM, New York, NY, USA, 2011, pp. 16:1–16:4. doi:10.1145/2019136.2019154.
URL http://doi.acm.org/10.1145/2019136.2019154

[16] W. Tsai, X. Sun, Saas multi-tenant application customization, in: 2013 IEEE Seventh International Symposium on Service-Oriented System Engineering, 2013, pp. 1–12. doi:10.1109/SOSE.2013.44.

[17] L. Ying, Z. Bin, L. Guoqi, W. Deshuai, G. Yan, Personalized modeling for saas based on extended wscl, in: 2010 IEEE Asia-Pacific Services Computing Conference, 2010, pp. 355–362. doi:10.1109/APSCC.2010.38.

[18] F. Gey, D. V. Landuyt, W. Joosen, Middleware for customizable multi-staged dynamic upgrades of multi-tenant saas applications, in: 2015 IEEE/ACM 8th International Conference on Utility and Cloud Computing (UCC), 2015, pp. 102–111. doi:10.1109/UCC.2015.26.

[19] H. Moens, F. De Turck, Feature-based application development and management of multi-tenant applications in clouds, in: Proceedings of the 18th International Software Product Line Conference - Volume 1, SPLC '14, ACM, New York, NY, USA, 2014, pp. 72–81. doi:10.1145/2648511.2648519.
URL http://doi.acm.org/10.1145/2648511.2648519

[20] Y. Shi, S. Luan, Q. Li, H. Wang, A multi-tenant oriented business process customization system, in: 2009 International Conference on New Trends in Information and Service Science, 2009, pp. 319–324. doi:10.1109/NISS.2009.181.

[21] T. Nguyen, A. Colman, J. Han, A feature-based framework for developing and provisioning customizable web services, IEEE Transactions on Services Computing 9 (4) (2016) 496–510. doi:10.1109/TSC.2015.2405546.

[22] H. Moens, E. Truyen, S. Walraven, W. Joosen, B. Dhoedt, F. De Turck, Developing and managing customizable software as a service using feature model conversion, in: 2012 IEEE Network Operations and Management Symposium, 2012, pp. 1295–1302. doi:10.1109/NOMS.2012.6212066.

[23] H. Moens, B. Dhoedt, F. De Turck, Allocating resources for customizable multi-tenant applications in clouds using dynamic feature placement, Future Gener. Comput. Syst. 53 (C) (2015) 63–76. doi:10.1016/j.future.2015.05.017.
URL http://dx.doi.org/10.1016/j.future.2015.05.017

[24] W. Liu, B. Zhang, Y. Liu, D. Wang, Y. Zhang, New model of saas: Saas with tenancy agency, in: 2010 2nd International Conference on Advanced Computer Control, Vol. 2, 2010, pp. 463–466. doi:10.1109/ICACC.2010.5486635.

[25] A. Rico, M. Noguera, J. L. Garrido, K. Benghazi, J. Barjis, Extending multi-tenant architectures: A database model for a multi-target support in saas applications, Enterp. Inf. Syst. 10 (4) (2016) 400–421. doi:10.1080/17517575.2014.947636.
URL http://dx.doi.org/10.1080/17517575.2014.947636

[26] S. T. Ruehl, H. Wache, S. A. W. Verclas, Capturing customers' requirements towards mixed-tenancy deployments of saas-applications, in: 2013 IEEE Sixth International Conference on Cloud Computing, 2013, pp. 462–469. doi:10.1109/CLOUD.2013.42.

[27] M. Makki, D. Van Landuyt, S. Walraven, W. Joosen, Scalable and manageable customization of workflows in multi-tenant saas offerings, in: Proceedings of the 31st Annual ACM Symposium on Applied Computing, SAC '16, ACM, New York, NY, USA, 2016, pp. 432–439. doi:10.1145/2851613.2851627.
URL http://doi.acm.org/10.1145/2851613.2851627

[28] I. Kumara, J. Han, A. Colman, T. Nguyen, M. Kapuruge, Sharing with a difference: Realizing service-based saas applications with runtime sharing and variation in dynamic software product lines, in: 2013 IEEE International Conference on Services Computing, 2013, pp. 567–574. doi:10.1109/SCC.2013.30.

[29] R. Mietzner, F. Leymann, M. P. Papazoglou, Defining composite configurable saas application packages using sca, variability descriptors and multi-tenancy patterns, in: 2008 Third International Conference on Internet and Web Applications and Services, 2008, pp. 156–161. doi:10.1109/ICIW.2008.68.

[30] W. Lee, M. Choi, A multi-tenant web application framework for saas, in: 2012 IEEE Fifth International Conference on Cloud Computing, 2012, pp. 970–971. doi:10.1109/CLOUD.2012.27.

[31] J. Schroeter, S. Cech, S. Goetz, C. Wilke, U. Amann, Towards modeling a variable architecture for multi-tenant saas-applications, in: Proceedings of the Sixth International Workshop on Variability Modeling of Software-Intensive Systems, VaMoS '12, ACM, New York, NY, USA, 2012, pp. 111–120. doi:10.1145/2110147.2110160.
URL http://doi.acm.org/10.1145/2110147.2110160

[32] I. Kumara, J. Han, A. Colman, M. Kapuruge, Software-defined service networking: Runtime sharing with performance differentiation in multi-tenant saas applications, in: 2015 IEEE International Conference on Services Computing, 2015, pp. 210–217. doi:10.1109/SCC.2015.37.

[33] A. Correia, J. R. Penha, A. M. R. da Cruz, An architectural model for customizing the business logic of saas applications, in: ICSOFT, 2013.

[34] N. K. Salih, T. Zang, Variable service process by feature meta-model for saas application, in: 2012 International Conference on Green and Ubiquitous Technology, 2012, pp. 102–105. doi:10.1109/GUT.2012.6344158.

[35] J. Müller, J. Krüger, S. Enderlein, M. Helmich, A. Zeier, Customizing enterprise software as a service applications: Back-end extension in a multi-tenancy environment, in: J. Filipe, J. Cordeiro (Eds.), Enterprise Information Systems, Springer Berlin Heidelberg, Berlin, Heidelberg, 2009, pp. 66–77.

[36] S. Aulbach, M. Seibold, D. Jacobs, A. Kemper, Extensibility and data sharing in evolving multi-tenant databases, in: 2011 IEEE 27th International Conference on Data Engineering, 2011, pp. 99–110. doi:10.1109/ICDE.2011.5767872.

[37] F. Aulkemeier, M. A. Paramartha, M.-E. Iacob, J. van Hillegersberg, A pluggable service platform architecture for e-commerce, Information Systems and e-Business Management 14 (3) (2016) 469–489. doi:10.1007/s10257-015-0291-6.
URL https://doi.org/10.1007/s10257-015-0291-6

[38] W. Sun, K. Zhang, S.-K. Chen, X. Zhang, H. Liang, Software as a service: An integration perspective, in: B. J. Krämer, K.-J. Lin, P. Narasimhan (Eds.), Service-Oriented Computing – ICSOC 2007, Springer Berlin Heidelberg, Berlin, Heidelberg, 2007, pp. 558–569.

[39] M. Almorsy, J. Grundy, A. S. Ibrahim, Tossma: A tenant-oriented saas security management architecture, in: 2012 IEEE Fifth International Conference on Cloud Computing, 2012, pp. 981–988. doi:10.1109/CLOUD.2012.146.

[40] T. Scheibler, R. Mietzner, F. Leymann, Eai as a service - combining the power of executable eai patterns and saas, in: 2008 12th International IEEE Enterprise Distributed Object Computing Conference, 2008, pp. 107–116. doi:10.1109/EDOC.2008.21.

[41] Y. Zhang, S. Liu, X. Meng, Towards high level saas maturity model: Methods and case study, in: 2009 IEEE Asia-Pacific Services Computing Conference (APSCC), 2009, pp. 273–278. doi:10.1109/APSCC.2009.5394111.

[42] W. Sun, X. Zhang, C. J. Guo, P. Sun, H. Su, Software as a service: Configuration and customization perspectives, in: 2008 IEEE Congress on Services Part II (services-2 2008), 2008, pp. 18–25. doi:10.1109/SERVICES-2.2008.29.

[43] M. Helmich, J. Müller, J. Krüger, A. Zeier, S. Enderlein, H. Plattner, Mappermania: A framework for native multi-tenancy business object mapping to a persistent data source, in: AMCIS, 2009.

[44] D. Ziani, A. AlShehri, A new framework for customizing erp systems in a multi tenant saas environment, in: 2015 2nd World Symposium on Web Applications and Networking (WSWAN), 2015, pp. 1–7. doi:10.1109/WSWAN.2015.7209089.

[45] L. Kong, Q. Li, X. Zheng, A novel model supporting customization sharing in saas applications, in: 2010 International Conference on Multimedia Information Networking and Security, 2010, pp. 225–229. doi:10.1109/MINES.2010.57.

[46] A. Khanjani, W. N. W. A. Rahman, A. A. A. Ghani, A. B. M. Sultan, Saas quality of service attributes, Journal of Applied Sciences 14 (24) (2014) 3613–3619. doi:10.3923/jas.2014.3613.3619.

[47] H. J. La, S. D. Kim, A systematic process for developing high quality saas cloud services, in: M. G. Jaatun, G. Zhao, C. Rong (Eds.), Cloud Computing, Springer Berlin Heidelberg, Berlin, Heidelberg, 2009, pp. 278–289.

[48] J. Y. Lee, J. W. Lee, D. W. Cheun, S. D. Kim, A quality model for evaluating software-as-a-service in cloud computing, in: 2009 Seventh ACIS International Conference on Software Engineering Research, Management and Applications, 2009, pp. 261–266. doi:10.1109/SERA.2009.43.

[49] P. Nadanam, R. Rajmohan, Qos evaluation for web services in cloud computing, in: 2012 Third International Conference on Computing, Communication and Networking Technologies (ICCCNT'12), 2012, pp. 1–8. doi:10.1109/ICCCNT.2012.6395991.

[50] A. Zia, M. N. A. Khan, Identifying key challenges in performance issues in cloud computing, International Journal of Modern Education and Computer Science 4 (10) (2012) 59.

[51] M. R. A. Akojwar, M. R. V. Kothari, M. S. A. Kahate, M. R. D. Ganvir, Software as a service with cloud computing, IJECCE 3 (1) (2012) 149–155.

[52] CSMIC, Service measurement index framework version 2.1 (July 2014).
URL http://csmic.org/downloads/SMI_Overview_TwoPointOne.pdf

[53] M. H. Cancian, J. C. R. Hauck, C. G. von Wangenheim, R. J. Rabelo, Discovering software process and product quality criteria in software as a service, in: Product-Focused Software Process Improvement, Springer Berlin Heidelberg, Berlin, Heidelberg, 2010, pp. 234–247.

[54] M. Alhamad, T. Dillon, E. Chang, Conceptual sla framework for cloud computing, in: 4th IEEE International Conference on Digital Ecosystems and Technologies, 2010, pp. 606–610. doi:10.1109/DEST.2010.5610586.

[55] M. Salama, A. Shawish, A. Zeid, M. Kouta, Integrated qos utility-based model for cloud computing service provider selection, in: 2012 IEEE 36th Annual Computer Software and Applications Conference Workshops, 2012, pp. 45–50. doi:10.1109/COMPSACW.2012.18.

[56] E. Badidi, A framework for software-as-a-service selection and provisioning, ArXiv abs/1306.1888.

[57] J. Song, S. Zhang, Y. Gong, B. Dai, A qos evaluation model for test-bed in the cloud computing environment, in: 2012 IEEE Ninth International Conference on e-Business Engineering, 2012, pp. 292–295. doi:10.1109/ICEBE.2012.54.

[58] Q. He, J. Han, Y. Yang, J. Grundy, H. Jin, Qos-driven service selection for multi-tenant saas, in: 2012 IEEE Fifth International Conference on Cloud Computing, 2012, pp. 566–573. doi:10.1109/CLOUD.2012.125.

[59] Shangguang Wang, Z. Zheng, Qibo Sun, Hua Zou, Fangchun Yang, Cloud model for service selection, in: 2011 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), 2011, pp. 666–671. doi:10.1109/INFCOMW.2011.5928896.

[60] A. Q. Ali, A. B. M. Sultan, A. A. Abd Ghani, H. Zulzalil, A systematic mapping study on the customization solutions of software as a service applications, IEEE Access 7 (2019) 88196–88217.
URL https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8747000

[61] A. Q. Ali, A. B. M. Sultan, A. A. Abd Ghani, H. Zulzalil, Development of a valid and reliable software customization model for saas quality through iterative method: perspectives from academia, PeerJ Computer Science.(2020) (in press). doi:10.7717/peerj-cs.294.