Preference-based Internet of Things dynamic service selection for smart campus

[1]Lindelweyizizwe Manqele, [2]Mqhele Dlodlo
Department of Electrical Engineering
University of Cape Town
Cape Town, South Africa
[1]MNQSIZ001@myuct.ac.za
[2]Mqhele.dlodlo@uct.ac.za

[1]Lindelweyizizwe Manqele, [3]Louis Coetzee, [4]Quentin Williams and [5]George Sibiya
Meraka Institute
CSIR
Pretoria, South Africa
[1]Lmanqele, [3]lcoetzee1, [4]qwilliams, [5]gsibiya{@csir.co,za}

***Abstract*— The usage of the Internet of Things technology across different service provisioning environments has increased the challenges associated with service discovery and selection. Users cannot always remember the Internet Protocol (IP) address for every service they need to utilize from the middleware registry. In order to address this challenge, an architecture that enables a representation of user preferences and manipulates relevant services description of available services is developed. This paper, an algorithm derived from the architecture that contributes towards addressing the service selection and discovery problem is proposed. The accuracy of the algorithm is evaluated based on response time, recall and precision metrics. The experiments show that the content-based algorithm works better than collaborative algorithm based on user preference. The content-based algorithm more returns relevant services to the user and takes shorter time as compared to the collaborative filtering.**

Keywords — Internet of Things, service discovery, service selection, recommender system, context-aware algorithm.

I. INTRODUCTION

The Internet of Things (IoT) appears to be the anticipated technology occurrence that will influence the future [3]. The IoT involves an increasing number of smart interconnected devices and sensors (e.g. cameras, biometric, smart meter, and medical sensors) for a smart world [11], these interconnected devices and sensors are referred as "Things". The IoT is not dependent on developing new technology but on connecting and integrating existing technologies. Peer-to-peer communications among devices will push services down to the device layer that implements "Things" and create new opportunities for functionality like discovery and selection [5].

Consequently, the selection of suitable services in IoT becomes more complex. As the number of devices on the network is increasing, so also is the number of services. In order to address the challenge of service selection, we need mechanisms to represent user preferences and manipulate relevant service descriptions of available services [7]. There are attempts by other researchers such as [1][2][6][9] to addresses service selection and discovery challenge that are later addressed in the literature review. However, the matchmaking on the literature becomes limited on few attributes of the service while a comprehensive user profile model can play a role in the recommendation of services more accurately.

This work investigates mechanisms that can be used to select IoT services within smart campus. According to the focus of this paper, the smart campus refers to an organisation that utilises services as provided by the IoT service provider. The IoT services storage becomes critical in the smart campus because of large data. The data has to be stored intelligently and be used for smart monitoring and actuation. Storing data intelligently means the data can be used for smart and automated decision making. However, Evans in [5] saw a need to develop artificial intelligence algorithms that could be centralised or distributed based on the need to make sense of the collected data. Furthermore, an approach emanating from recommender system is suitable to be used in smart campus environment because recommender systems pre-select services a user might be interested [9].This paper presents a mechanisms that can be used to address the challenge of service selection in smart environments.

The paper is organised as follows: section II discusses existing approaches. Section III presents an architecture as part of proposed solution towards service selection. Section IV proposes the algorithm derived from the architecture proposed in section III. Section V presents application scenario. Section VI discusses the results for the algorithm performance. Section VII summarizes the work and draws conclusion. Lastly, section VIII presents the future work.

II. LITERATURE REVIEW

Theoretical evaluation components of service selection based on the application scenario approaches is briefly discussed as follows: *Domain-specification* that gives an awareness to support the dynamic environments like smart campus. Smart environments are based on ubiquitous computing where environments interact with its inhabitants on a device layer [12]. *Storage* – a domain like smart campus is likely to have distributed resources and services. However, there is a need to get a paradigm that will be able to integrate services logically to make the use of service selection possible. From the literature review conducted, cloud storage is centralized database where data is stored in virtualised pools of storage which are generally hosted by third parties [19].

Scalability refers to the ability of a system, network, or process to handle a growing amount of services in a capable manner. IoT is a global network infrastructure linking physical

and virtual objects through the exploitation of data capture and communication capabilities [13]. Therefore, such an environment demands large data storage and sharing of resources and services. *Algorithm* to be used should support recommendation since the selection of services will be done according to the need. Users save time by using recommender system that helps them to choose from variety of options. The purpose of recommender systems is to pre-select information a user might be interested in [7].

Evaluation of an architecture should be tested by using the architecture to select services. It is important to check the relationship between the components and how they perform together. *Quality of Service (QoS)* is a non-functional property of system. According to [16], those non-functional properties includes the service tracking which is a broker that has a service repository to record all feasible web services it is aware of. QoS-based broker selects services to execute a business process so that the user-defined utility is maximized and user's QoS requirements are satisfied. *User profile* captures a user's data and compiles user profiles. User profile helps to pre-select the services by matching user profile attributes with service profile attributes.

Table 1 summarizes possible approaches for service selection as published on the literature. The table breaks down each algorithm in terms of applicable criteria. The summary leads to the theoretical evaluation of service selection approaches based on the application scenario. The scenario is as follows:

"The user is looking for a service that can do function Z. The user may request for this service within or outside the premises of the smart campus. The user expects the application to be able to predict and suggest services to the user."

TABLE I. THEORETICAL ARCHITECTURAL EVALUATION

Author(s)	User profile	Domain	Storage	Algorithm	Scalability	Evaluation	Quality of Service
(D'Mello, Ananthanarayana, & Thilagam, 2008) in [2]	X	√	√	X	X	√	√
D'Mello & Antony, 2010 in [1]	X	√	√	X	√	√	√
(Prabhakar T & Manikrao U, 2006) in [9]	X	X	√	X	X	√	√
Yu & Lin, 2006 in [15]	X	√	√	√	√	√	√
Guha ,2009 in [6]	X	√	√	√	√	√	√

D'mello in [2] and [1] proposed the semantic broker based web service architectures that recommended the best match for the requester based on the requested functionality, quality and business offerings. However, D'mello architecture does not use non-functional requirement as an input to improve the accuracy of the semantic broker. Prabhakar and Manikrao in [9] proposed an architecture of web service selection framework using recommendation system based on user feedback and collaborative filtering techniques. The challenge with Prabhakar and Manikrao's architecture is cold start; it experiences challenge of recommending new services on the database.

The evaluation shows that none of the service selection architectures considered to model user profile in order to make the selection more accurate by comparing more service attributes. Recommender and registry components of Yu & Lin, 2006 in [16] and Guha ,2009 in [6] architectures contributed to this paper. Hence, section III presents the proposed architecture.

III. PROPOSED ARCHITECTURE

This section presents the proposed architecture and briefly discusses each of the components. As demonstrated in Figure 1 the following components are applicable.

A. Application layer

Application layer presents three sub-components components that are accessible to the user and those components are briefly discussed as follows: *User registration* is a user interface that registers the records for a user by completing required fields. It helps in identifying the user and trace a user's usage history and user's preferences. *Service search* is a user web interface sub-component that allows a user to define a service the user is looking for. *Service View* gives the list of recommended service(s) based the attributes used on service search. *User View* allows a user to choose a service among recommended services and invoke the service.

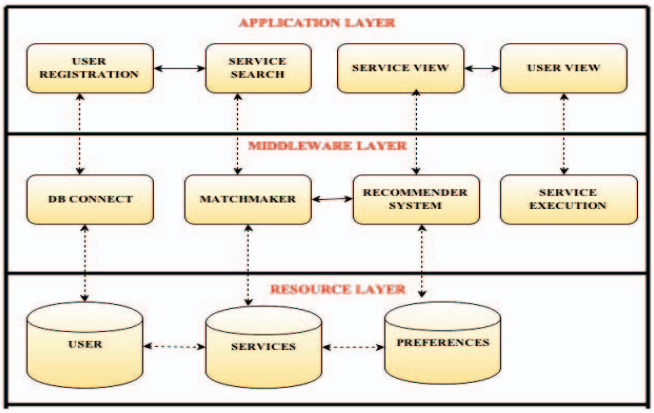

Fig. 1. IoT service selection architecture

B. Middleware layer

Middleware layer presents four sub-components that orchestrate the service selection that are briefly discussed as follows: *DB Connect* connects database stored in registries to be utilises by the user. *Matchmaker* lists available services based on the user context (location, time). System compares the user profile compiled by "service search" with available service profile and recommend the services based on the match

between profiles and preferences. *Service execution* invokes the recommended services.

C. Resource layer

Resource layer presents three sub-components that stores the database of user, preferences and services. *User database* keeps the records about the user (forms part of user profile). *Resource database* keeps records about resources and Things. This sub-component keeps real-time records, it keeps records updated ready to be utilised. *Preference database* keeps records about what a user prefers and usage history. The next section we present the algorithm.

IV. PROPOSED ALGORITHM

In this section we present an algorithm that matches a service profile and user profile. The algorithm is initiated by modelling the user profile and IoT service profiles using set theory. After the profile modelling, the matching process of the two profiles are presented

A. User Profile and Service Profile Modeling

We consider the user to be having both dynamic and static profile attributes. Examples of static user profile attributes include among others *name, surname and home address or home_location*.

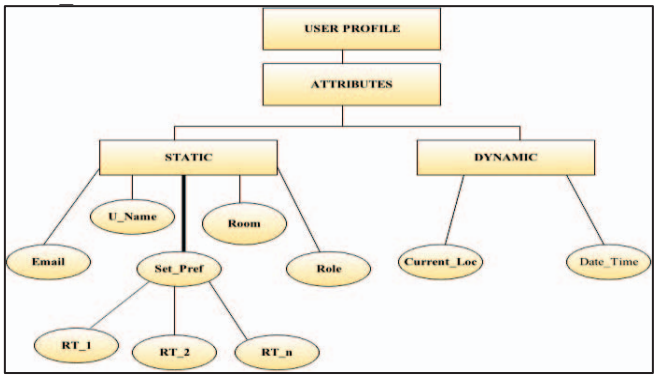

Fig. 2. UserProfile

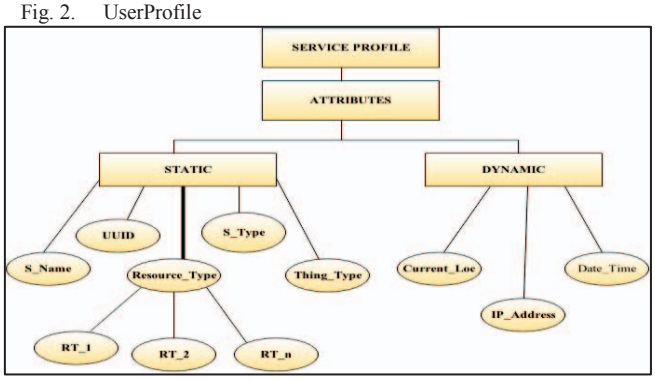

Fig. 3. Service Profile

On the other hand, dynamic user profile attributes include aspects such as *current location*. The user profile is as depicted in Figure 2. For discovery of a service, the user profile has to be matched against IoT service profiles. The service profile is as depicted in Figure 3. To generalize the user profile and the service profiles make use of mathematical models. Let the set of static attributes that can be used to represent a user profile be A_{stat}. Therefore,

$$A_{stat} = \{x | x \text{ is a static user attribute}\} \quad \text{Eq. 1}$$

Since user profile also comprises of dynamic attributes, we represent the set of dynamic attributes as A_{dyn}.

$$A_{dyn} = \{y | y \text{ is a dynamic user attribute}\} \quad \text{Eq. 2}$$

The user profile is therefore made up of the two sets of attributes. i.e.,

$$P_u = A_{stat} \cup A_{dyn} \quad \text{Eq. 3}$$

The service as well has both dynamic attributes and static attributes. In this paper, we consider an IoT service to be a web service endpoint which has a direct access to one or more IoT resources [13] . Such a service has a set of dynamic attributes and static attributes which we represent as B_{dyn} and B_{stat} respectively. The two sets are then represented as follows;

$$B_{dyn} = \{q | q \text{ is a dynamic service attribute}\} \quad \text{Eq. 4}$$

And,

$$B_{stat} = \{r | r \text{ is a static service attribute}\} \quad \text{Eq. 5}$$

The service profile comprises of the union of the two sets, i.e.,

$$Ps = B_{dyn} \cup B_{stat} \quad \text{Eq. 6}$$

During the recommendation process, the matchmaking algorithm determines similarities between the user profile set, Set P_u in equation above and service profile set, Set P_s. It is worth noting that in this paper we consider x, y, q and r to be of both numerical and textual value types. The numerical values are further categorized as of continuous numerical value types and discrete value types. For example, distance, if a continuous value type is valid while a number of resources on a Thing are of a discrete nature.

For example if,

$$x = \{\{x_i | x_i discrete\}, \{x_j | x_j \text{ continuous}\}\} \quad \text{Eq. 7}$$

Then becomes,

$$P_u = \{\{\{x_{i,k}\}, \{x_{j,l}\}\}, \{y\}\} \quad \text{Eq. 8}$$

Similarly, the service profile in Eq. 6 can be given by:

$$P_s = \{\{\{q_{i,k}\}, \{q_{j,l}\}\}, \{r\}\} \quad \text{Eq. 9}$$

It should be noted that as part of the user profile, there are preferences that are specified by the user during the enrolment or registration. The preferences are basically service attribute name-value pairs that a user specifies. They are, therefore, a subset of attributes that are prioritized during the matchmaking process. In the next section it will be discussed in details how this subset is taken into consideration during matchmaking process.

B. *Profile matching*

Having formally expressed both the service and the user profile, the matchmaking algorithm that enables the recommendation process will be discussed. Recommenders systems are the computer-based intelligent techniques to deal with problem finding appropriate services from large number of available services. i.e. it is a function that takes P_u and P_s as inputs and produces a matching score. The matching score is what is used to rank services for returning to the user. In this section we discuss how these profiles are matched for recommendation of the services based on profiles and preferences of the user.

In essence, matchmaking utilizes a function that aggregates the matching scores of each attribute's values from the service and the user profile. There is a unique function for each attribute which determines similarity of the specific attribute values from both the service profile and the user profile. As an example, a function that determines the vicinity of a service to the user (distance) may not be used to compare resource types specified in the user preferences set against the resource type of the service being matched. A different function that compares the resource types is therefore required. The matching function therefore would be:

$$F(P_u, P_s) = \sum_{i=0}^{n} w_i f_i(P_u, P_s) \qquad \text{Eq. 10}$$

where n, is the total number of attributes to be matched, w_i is the assigned weight of attribute a_i and f_i is the function used to compare values of attribute a_i.

Step 1. The user preferences set is considered. This set of name-value pair attributes needs to be compared against the attribute name-value pairs of the published IoT services. Often, one of the attributes that will be specified during enrolment is the resource types set. The output from this step will be set of attribute name-value pairs. With this set of attribute name-value pairs.

Step 2. The second phase of the matchmaking algorithm involves matching the user profile and the service profile based on the attribute name-value pairs that were not specified in the user preferences. Retrieve user profile attribute name-value pairs not specified in preferences.

Step 3. Match each service profile against the user profile.

Step 3.1. In this step, the attributes in both profiles are compared and the ones that are common among the profiles are matched. For example, if the user has his current location specified in his profile and the service which may be mobile has current the location specified, the current location attribute is included in the set of attributes against which the service and the user profile will be matched against. This involves selecting a set of functions $\{f_i\}$ with each function corresponding to an attribute to be matched. An output from this step is a set of attribute names.

Step 3.2. Proceed to Step 3.4.

Step 3.3. Match non-common attributes between user profile and the service profile based on semantics. e.g. attributes, "permanent location" and "home location" are semantically similar.

Step 3.3.1. Get semantically similar attribute names from service profiles and user profile.

Step 3.4. If the list of services to match is not empty proceed to Step 3.4.1. else proceed to Step 4.

Step 3.4.1. For each attribute name-value pair from the user profile is matched against the attribute name-value pair of the service profile. The output from this step will be a set of services that matched based on these attributes. This is where Eq. 10 is applied. A service is considered to have some matching degree if and only if:

$$F(P_u, P_s) > 0$$

Step 3.4.2. If results are not returned and the attribute is of a dynamic and numerical nature and not at maximum value, the thresh hold is increased and the algorithm returns to Step 3.4.1.

Step 3.4.3. If results were found and are more than the maximum desired list length and the attribute is of a dynamic and numerical nature and not at minimum value, the thresh hold is reduced and the algorithm returns to Step 3.4.1.

Step 3.4.4. If results are not found, all dynamic attributes are at minimum and non-preferred attributes are not considered, the algorithm proceeds to Step 2.

Step 3.5. Rank services. The services are sorted based on their computed value of the function $F(P_u, P_s)$.

Step 3.6. Return recommended list

Step 4. Notify user and Exit

The next section presents application scenario.

V. APPLICATION SCENARIO

Assume that user is looking for service that can do Z. The user will be required to create an account. The user's account will compile profile while service profile already exist from service registry. The results will be as indicated in Figure 4

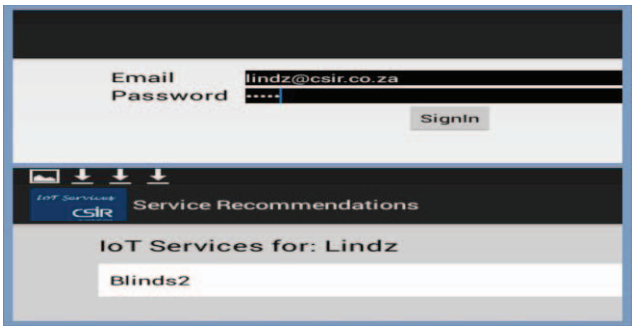

Fig. 4. IoT services App

The application then recommends services that the user can utilize based on user preferences and role. The application was implemented on Android Studio version 0.8.2.The application was installed and tested on android device Galaxy S4 GT-19500 that utilizes. SQLite to store users authentication information. In Figure 4, the user is requesting the service(s) by logging and the application recommend service(s) to the user. The next section presents the results analyses.

VI. RESULTS ANALYSIS

The evaluation was conducted using response time, recall and precision metric to measure performance and effectiveness of the algorithm on the mobile device [18]. The evaluation was compared between content-based and collaborative filtering as the candidates remained based on recommender systems classification conducted by Manqele *et al.* in [17]. Content-based recommender algorithm recommends services based on user preferences only while collaborative filtering recommend services based on what other users liked before.

A. Performance

The performance of the application was measured by the response time. The total time taken by the application from request to the recommendation shows that content-based was more effective than the collaborative filtering.

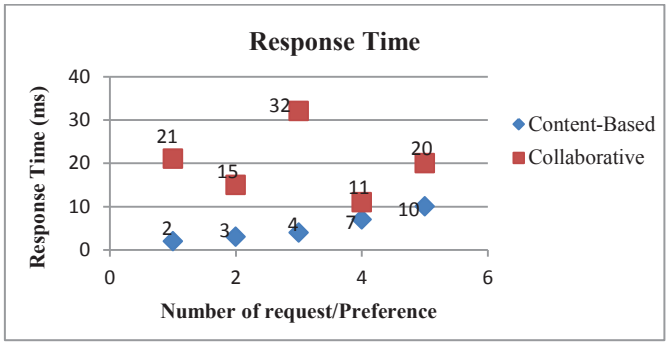

Fig. 5. Response Time for content-based

B. Effectiveness

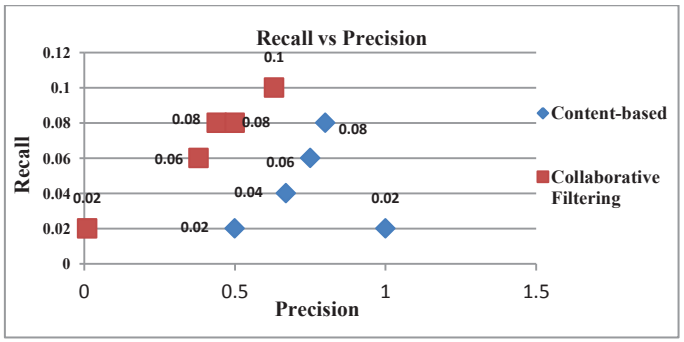

Fig. 6. Recall vs precision

Preference recall measures how good the architecture developed in making sure that there is no missing relevant recommendations. Preference precision measures how good the framework is in reducing irrelevant recommendations. A good preference model is expected to optimise these two parameters. The results show that content-based is more effective than collaborative filtering technique. The results are shown in Figure 5 and Figure 6.

VII. CONCLUSION

The user profile and service profile had to be modelled. The proposed algorithm selects services based on user preference and user context. The evaluation of the algorithm based response time, recall and precision metrics yielded the results. The results showed that content-based technique works better than collaborative filtering in both effectiveness and performance.

VIII. FUTURE WORK

The modelling of user and service profile created an opportunity to match service(s) using semantic match. Other researcher may not need users to define their preferences but learn user's behaviour and recommend services based on history usage.

References

[1] D'Mello, D.A.,2010. Semantic Web Service Selection Based on Service Provider ' s Business Offerings. , 10(2), pp.25–37.

[2] D'Mello, D.A., Ananthanarayana, V.S. & Thilagam, S., 2008. A QoS Broker Based Architecture for Dynamic Web Service Selection. *2008 Second Asia International Conference on Modelling & Simulation (AMS)*, pp.101–106. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber*r=4530459 [Accessed April 16, 2014].*

[3] *Dlodlo, N.Foko, T., Mvelase, P. & Mathaba, S., 2012. The State o*f Affairs in Internet of Things Research. , 15(3), pp.244–258.

[4] *Dutton, W.H., 2005. The Internet of* Things. *ITU INTERNET REPORT 2005*.

[5] *Evans, D., 2011. The Internet of Thi*ngs How *the Next Evolution of th*e *Internet The Internet of Things How* the Nex*t Evolution of the Inter*net *Is Changing Everything.* , (April).

[6] *Guha, 2009. Investigation of Service Selection Algorithms for Grid Services A Thesis Submitted t*o the College of Graduate Studies and Research *in Partial Fulfillment* of the Re*quirements for the degree of Master* of Scienc*e in the Division of Comput*er *Science Universit.* , (October).

*[7] Manikrao, U.S. & Prabhakar, T. V, 2005. Dynamic Selection of Web Services with Recommendation Sy*stem. In *In: Proceedings of the International Conference on Next Generation Web Services Practices (NWESP), IEEE Computer Society*. Press, p. 117.

*[8] Pilgrim Beart, 2013. Internet of Thi*ngs. , (*February 2012).*

*[9] Prabhakar T & Manikrao U, 2006. Dyna*mic sele*ction of web services with Recommendation system. In Kanpur: Indian Institute of Technology. Available at: http://www.cse.iitk.ac.in/users/tvp/papers/Umardand-Dynamic.pdf Last accessed April 2012.*

[10] *Samreen, S.U. and N.A.* and A., 2011. *Dynamic Service Composition in SOA and QoS Related Issues*.

[11] Serbanati, A., Maria, M.C. & Biader, C.U., 2011. Building Blocks of the Internet of Things : State of the Art and Beyond.

[12] Silva, L. C., Costa, C. A., Geyer, C., Augustin, I., & Maria, S. (2008). On the Control of Adaptation in Ubiquitous Computing (pp. 2228–2229).

[13] Sundmaeker, H. & Saint-exupéry, A. De, 2010. *Vision and Challenges for Realising the Internet of Things*,

[14] Thoma, M., Braun, T. & Magerkurth, C., 2014. Enterprise Integration of Smart Objects using Semantic Service Descriptions.

[15] Wang, H., 2013. Toward a Green Campus with the Internet of Things – the Application of Lab Management. , II, pp.1–5.

[16] Yu, T. & Lin, K., 2006. A Broker-Based Framework for QoS-Aware Web Service Composition. *2005 IEEE International Conference on e-Technology, e-Commerce and e-Service*, pp.2229. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1402263.

[17] Manqele S., Dlodlo N., Xulu S. & Adigun M, 2012, Selection and provisioning of services in cloud using recommender systems approach for SMME.

[18] Jesse D., 2007 , The Relationship between Precision-Recall and ROC curves.

[19] Sukhamrit, K., Kuljit, K., & Dilbag, S. (2012). Evaluating Performance of Web Services in Cloud Computing Environment withHigh Availability. *Global Journal of Computer Science and Technology: B Cloud & Distributed*, *12*(11).