A new approach for service discovery and prediction on Pervasive Information System

Salma Najar, Manuele Kirsch Pinheiro, Carine Souveyet

Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia Computer Science00 (2014) 000–000

Procedia Computer Science

www.elsevier.com/locate/procedia

5th International Conference on Ambient Systems, Networks and Technologies (ANT-2014), the 4thInternational Conference on Sustainable Energy Information Technology (SEIT-2014)

A new approach for service discovery and prediction on Pervasive Information System

Salma Najar*, Manuele Kirsch Pinheiro, Carine Souveyet

Centre de Recherche en Informtique / Université Paris 1 – Panthéon Sorbonne, 90 Rue Tolbiac, Paris 75013, France

Abstract

Recent evolution of technology transformed the way we interact with Information Systems (IS), leading to a new generation of IS, the Pervasive Information Systems (PIS).These systems have to face heterogeneous pervasive environments, whose complexity they must hide from end-user. In order to reach transparency and proactivity necessary for successful PIS, new discovery and prediction mechanisms are necessary. In this paper, we propose a new user-centric approach for service discovery and prediction onPIS based on both user's context and intentions. Intentions allow focusing on goals user wants to satisfy when requesting a service. Those intentions rise in a given context, which may condition the service implementation. We propose then a service discovery mechanism that observes user's context and intention in order to offerhim the service that may best satisfy her/his intention on the current context. We also propose a prediction mechanism that tries to anticipate user'sintentions considering the observed context and user's history.

Selection and peer-review under responsibility of Elhadi M. Shakshuki.

*Keywords:*pervasive information system; service oriented computing; intention; context-aware computing; service discovery; service prediction

1. Introduction

New technologies transform the way we interact with IS and the services they offer, expanding the frontiers of IS outside the companies' environment. The BYOD (Bring Your Own Device) illustrates this tendency: employees bring their own devices to the office and keep using them to access the IS even when they are on the move. The

* Corresponding author. Tel.: +33-144-078-604; fax: +33-144-078-654.
E-mail address: salma.najar@malix.univ-paris1.fr

Selection and peer-review under responsibility of Elhadi M. Shakshuki.

consequence of such evolution is that IS now have to cope with a pervasive environment and may integrate services from very different natures. A new generation of ISis then rising, the Pervasive Information Systems (PIS).

Pervasive Information Systems intend to increase user's productivity by making IS services available anytime and anywhere. These systems changed the interaction paradigm from desktop computing to new technologies.They evolved from a fully controlled environment (the office) to a dynamic pervasive one[1]Contrary to traditional IS, PIS have to support a multitude of heterogeneous device and service types, challenging its design. We argue that PIS should be designed for helping user to better satisfy her/his needs according to her/his environment. PIS must not only consider the goals it must reach as an IS, but also handle pervasive environment heterogeneity. Indeed, it should hide this heterogeneity from the user, allowing her/him to concentrate on her/his needs, instead of on the technology itself. Transparency and proactivity become then key aspects onPIS, whichrequires offering user appropriate services consideringher/his goals and the context in which such goals appear, as well as the capability of anticipating future goals in this context. New service discovery and prediction mechanisms are then necessary.

We propose a new user-centric approach for service discovery and prediction considering PIS. This approach is based on both user's intentions, representing the goals she/he wants to achieve without saying how to perform it[2],and on the context in which these intentions have been formulated. The notion of context can be seen as any information that can be used to characterize the situation of an entity[3]. We consider that context information can influence service execution and, consequently, what service can be chosen to satisfy a given intention. In our opinion, both concepts should be considered during service discovery, since the main purpose is providing user with a service that can fulfill his goals in a fairly understandable and non-intrusive way. Thus, we propose a new service discovery mechanism that intendsdiscovering the available service that can satisfythe immediate user's intention in the current context. Based on the discovery results, we propose a newprediction mechanism that identifies common situations representing usage patterns, *i.e.*, recurrent context and intentions observed during PIS usage. By analyzing thesepatterns, the prediction mechanism learns user's behavior when using a PIS, and therefore anticipates future intentions and the most appropriate services that may satisfy it.

This paper is organized as follows: Section 2 presents related works on service discovery and prediction. Section 3 introduces the proposed service discovery mechanism, while Section 4 presents the service prediction mechanism. Section 5 presents an evaluation of the proposed mechanism, before concluding in Section 6.

2. Related works

During the last decade, a lot of research has been conducted concerning pervasive systems, mainly on context-aware services[4,5,6]Context-awareness becomes a necessary feature for providing adaptable services, for instance when selecting the best-suited service according to the relevant context information or when adapting the service during its execution according to context changes[7]. Different service discovery mechanisms have been proposed in the literature[5,6,8]. Most of them consider context information as a non-functional aspect of service[5,8], or as a condition for service selection and execution[6]. On both cases, a matchmaking, using semantic matching[8] or similarity measures[6], is performed between context information related to the service and the one related to user or execution environment. Nevertheless, only a few research works[9,10,11]consider the notion of intention during the service discovery. Intentions can be associated with service descriptionsas a set of capabilities, with their pre- and post-conditions[9]. They also can be used as a guide for service discovery, by a refining process in which intentions are decomposed on low-level intentions[10]. Unfortunately, the influence of context on the intention satisfaction is merely considered on the literature, context being often seen as a simple input on intention-based mechanisms[11].

A similar situation can be observed when considering service prediction. Even if several works have considered context prediction[12,13] or service recommendation based on context information[14,15], at the best of or knowledge, none has proposed combining intention aspect with context information. In the one hand, several works propose anticipating context information based on historical data[12,16] or pattern matching[13]. In the other hand, service prediction works proactively propose services based on user's historical context information. Most of these works[14,15,17] consider the correlation between context information and an item (*e.g.* a service) using different filtering techniques[17], sometimes correlated with ontology-based matching[15]. Unfortunately, the notion of intention, representing concrete user's goals, remains unexplored by these works.

3. Service Discovery Mechanism

In our user-centric approach, we propose a service discovery mechanism guided by user's intention and context. Its objective is to hide implementation complexity, and consequently to achieve the transparency promised by PIS. This service discovery selects the most appropriate service for the user, *i.e.*, the service that satisfies his/her immediate intention in a given context. It is based on a semantic service description and on a semantic matching algorithm. The goal of this algorithm is to rank the available services based on their contextual and intentional information. It compares semantically the user's intention with the intention that the service satisfies and user's current context with the service's context conditions. Then, the service having the highest matching score is selected. It represents the most appropriate service that satisfies user's immediate intention in his current context.

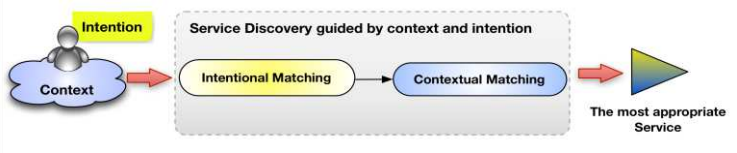

Fig. 1. Schematic view of the Services Discovery mechanism

The semantic matching algorithm, as illustrated in Fig. 1, is a two-step process: *intention matching* and *context matching*. In the first step, the ***intention matching*** is based on the use of ontologies and a semantic matching. Indeed, an intention can be representedasan-upletby a verb ($\mathcal{V}$) and a target ($\mathcal{T}$) representing user's goals[2,18].In this step, we propose to compare theuser's required intention $I_{\mathcal{U}} = <\mathcal{V}_{\mathcal{U}}, \mathcal{T}_{\mathcal{U}}>$ and the service's intention $I_{svi} = <\mathcal{V}_{svi}, \mathcal{T}svi>$ in two separated matching process.For the *verb matching*, we use a verb ontology ($Onto\mathcal{V}$), which contains a domain-specific set of verbs, representing significant actions authorized by the PIS, withtheir different meanings and relations. The degree of similarity is calculated based on the distance between these verbs in the verb ontology: (1/ L + 1), where L represents the number of links between two concepts in the ontology.We define five levels of similarity, inspired from[19], as illustrated inTable 1 (more details about the intention matching are presented in [20]).

Table 1. Verb Matchingrelations

Matching Relation	Explanation	Link	Score
Exact	Required verb is equivalent to the provided verb	0	1
Synonym	Required verb share a common signification with the provided verb	-	0,9
Hyponym	Required verb is more specific than the provided one	L	1/(L+1)
Hypernym	Required verb is more generic than the provided one	L	1/(L+1)
Fail	No relation between the two verbs	-1	0

Similarly, for the *target matching*, we use a domain-specific ontology, namely target ontology ($Onto\mathcal{T}$). This ontology represents the possible targets that are made available through the PIS. We compare the required target $\mathcal{T}_{\mathcal{U}}$ and the provided target $\mathcal{T}_{svi}$ using a degree of similarity also based on the distance between these concepts in the ontology. This semantic similarity, based on the works of [19], uses four levels: *exact*, *plugin*, *subsume* and *fails*. The *plugin* relation is similar to the *hyponym* relation in the verb matching, while the *subsume* relation is similar to the *hypernym* relation.

In the second step, the ***context matching***is based on a context ontology ($Onto\mathcal{CX}$) and a set of similarity measures. It matches individually the different context elements constituting the user ($\mathcal{CX}_{\mathcal{U}}$) and service context descriptions ($\mathcal{CXR}_{svi}$). Indeed, context information is often semantically represented using ontologies in which context information is structured (*e.g.*[5,8,16]). In our case, context information is representedas context elements (location, available memory, user's expertise, etc.) that are observed from a given subject (a user, a device, etc.). Both contextelements and subjects are semantically described using ontologies. Thus,the context description for a user ($\mathcal{CX}_{\mathcal{U}}$) represents a set of context observations $\mathcal{CX}_{\mathcal{U}} = \{cx_j\} \mid j>0$associated to this user and the context description for a service ($\mathcal{CXR}_{svi}$) represents a set of context conditions $\mathcal{CXR}_{svi} = \{cx_i\} \mid i>0$ expressed over context elements. The former corresponds to

the current observed user's context, while the latter represents contextual conditions for which service was designed (*i.e.* context under which service is able to better satisfy its intention).

The *context matching* score Cx_{score} is calculated as the sum of the scores of each context condition, as follows:

$$\boldsymbol{Cx_{score}} = \sum_{i=1}^{n} (\boldsymbol{w} * \boldsymbol{ContextConditionMatching}\ (\boldsymbol{cx_i}, \boldsymbol{cx_j}))$$

Thus, in order to define the relation *ContextMatching*, we consider the relation *ContextConditionMatching* that matches individually the user's context observations ($CX_{\mathcal{U}}$) and the service's context conditions (CXR_{Svi}). The context matchingproceeds as follows: for each conditioncx_i and observation cx_j, we first match the corresponding subjects, using the context ontology. This match is calculated, like the verb and target matching, based on the distance between both concepts in the ontology. If the matching score between them is higher than a given threshold then we match their context elements. This last matching takes also into account the weight assigned to it. Then, if the matching score between them is higher than a given threshold, only at this moment we evaluate the satisfaction of the context condition regarding to the user's context observations value one by one. More details about the context matching are presented in[20].

The observable context elements can be divided into several types. Context information values are distinguished between numerical or non-numerical types. In order to take into account this diversity, the relation *ContextConditionMatching*identifies the nature of the context element value and accordingly triggers the suitable measure to compare them. This relation evaluate if the user's context element satisfies the service's context element conditions, based on a specific operator (equal, not-equal, between, higher-than, etc.). For example, we have as a service's context condition that the device bandwidth must be higher than 12500. From the user's current context, if the captured value of the user's device bandwidth is really higher than 12500, then we return an exact match.

Besides, the weight (w) that the user allocates to each context attribute and whose value is between 0 and 1, represents the importance of an attribute to a given entity. The purpose here is to highlight the real importance of a context attribute according to user's preferences, and the importance of the attribute is proportional to its weight.

4. Service Prediction Mechanism

We propose also in this approach to predict the user's future intention. This approach recommends proactively a service that can fulfill user's future needs. It is based on the assumption that common situations (S_i) can be detected, even in a dynamic PIS. We define the notion of situation (S_i) as the user's intention ($I_{\mathcal{U}i}$), in a given context ($CX_{\mathcal{U}}$), satisfied by a specific service (Sv_i) resulting from a previous discovery process: $S_i = <I_{\mathcal{U}}, CX_{\mathcal{U}}, Sv_i>$. These situations are time-stamped and stored in a database after each service discovery process (history). Let the user's history $\mathcal{H}$be defined as a set of all the observed situations $\mathcal{S}$ ordered according to their time of occurrence. Thus, by analyzing the history ($\mathcal{H}$), the prediction mechanism can learn the user's behavior model ($\mathcal{M}c$) in a dynamic environment, and thus deduce its coming immediately intention.

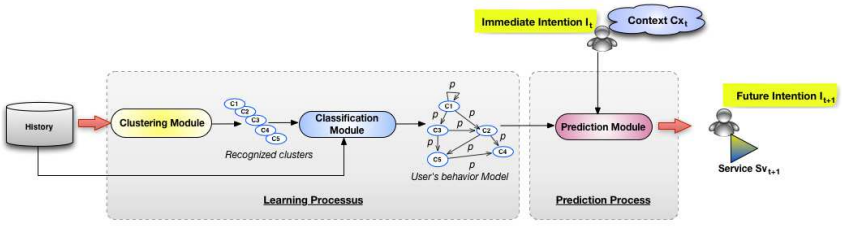

Fig. 2 Schematic view of the Services Prediction mechanism

Two main processes compose this service prediction mechanism: the *learning process* and the *prediction process*, as illustrated in Fig. 2. Thus, to realize anticipatory and proactive behavior of PIS, we need first to dynamically learn about the user and his behavior in a frequently changing environment. This represents the *learning process* where similar situations ($\mathcal{S}$) are grouped into clusters, during the phase of *clustering*. In the next step, these clusters are interpreted as states of a state machine. This phase is called *classification*. It aims to

represent, from the recognized clusters, the user's behavior model ($\mathcal{M}c$) based on his situations ($\mathcal{S}$). By interpreting situation changes as a trajectory of states, we can anticipate his future needs. Therefore, the *prediction process* is based on the user's behavior model ($\mathcal{M}c$), on the current user's intention ($I_{\mathcal{U}}$) and on the current user's context ($C_{\mathcal{X}}$).

More specifically, the main task of the *clusteringphase*, as illustrated inFig. 3.a, is to detect, for a given situation, the closest set of situations corresponding to highly similar intentions in quite similar context. This provides us a powerful mechanism to evaluate the user's intention. Indeed, a user can express the same intention in a slightly different way by using verbs and targets that are semantically similar enough. Based on verb and target ontologies, we perform a semantic matching between two intentions in order to determine their degree of similarity. The same applies to context information, since an intention may rise on similar contexts. More precisely, the input of this *clustering phase* represents vectors representing user's situations stored in the history. The main role of clustering is to detect recurrent situations among those previously observed and grouped in a *cluster*. A cluster consists of a *centroid* and a set of situations. The centroid represents the identifier of the cluster, which symbolizes the situation the most similar to all the situations grouped in this cluster. It is defined by the triplet $<I_{\mathcal{U}}, C_{\mathcal{X}}, \mathcal{S}v_i>$. After a clustering phase, the corresponding cluster identifier is attached to each new situations stored in the history.

In order to represent a situation, we attach a particular service to the couple <*Intention*, *Context*>. We believe that this represents a strong constraint (the concept situation is necessarily coupled to a particular service), but it opens a significant performance advantage, since it is not required to launch the service discovery mechanism during the prediction process. Thus, it is important to regularly update the clusters in order to have the service that best meets the couple intention and context of the situation.

Then, from these recognized clusters and the user's history, the *classification phase* determines and maintains a user's behavior model, as illustrated inFig. 3.b. This model represents the user's behavior as a set of states with a transition probability. Each state is represented by the centroid of the recognized cluster. Each probability is calculated based on the history and determines the probability of moving from one state to another.

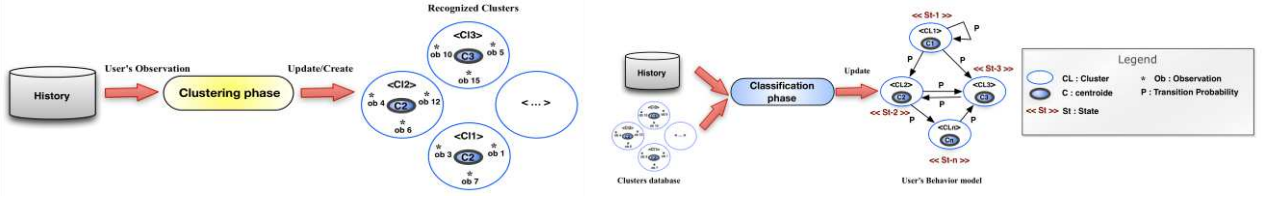

Fig. 3 a) The clustering phase, b) The classification phase

Several classification techniques exist. Among these techniques, the Markov chain[21]represents one of the well-known classification algorithms that can be used in a PIS. It represents a method for representing a stochastic process in discrete time with discrete state space. We represent the Markov chains model ($\mathcal{M}c$) as the doublet $\mathcal{M}c = (\mathcal{S}t, p)$, with $\mathcal{S}t$ representing the different states and $p \in [0,1]$ the probability of transition from one state to another.

At a given time *t*, the user is in a state $\mathcal{S}t_i$. In PIS, the user's intention and/or his context may change. Therefore, the user moves from the state $\mathcal{S}t_i$ to $\mathcal{S}t_j$. The state $\mathcal{S}t_j$ represents the successor of $\mathcal{S}t_i$ with a certain probability *p*. This transition probability represents the ratio of the transition from $\mathcal{S}t_i$ to $\mathcal{S}t_j$ divided by the number of all the possible transitions from $\mathcal{S}t_i$. This probability is represented as follow:

$$p_{\mathrm{St}_i\mathrm{St}_j} = P(\mathrm{X}_{t+1} = \mathrm{St}_j | \mathrm{X}_t = \mathrm{St}_i) = \frac{\mathrm{N}_{\mathrm{St}_i\mathrm{St}_j}}{\mathrm{N}_{\mathrm{St}_i\mathrm{St}_k}}$$

For each new situations stored in the history, we proceed by selecting for each situation $\mathcal{S}t_i$ (identified by his cluster), his successor $\mathcal{S}t_j$. The successor $\mathcal{S}t_j$ represents the situation that is directly stored after the situation $\mathcal{S}t_i$. Then, we calculate the number of the existing transitions from $\mathcal{S}t_i$ to $\mathcal{S}t_j$($\mathrm{N}_{\mathcal{S}t_i\mathcal{S}t_j}$). Next, for each situation $\mathrm{St_i}$ we determine the entire possible next situation $\mathrm{St_k}$($\mathrm{N}_{\mathcal{S}t_i\mathcal{S}t_k}$). We note that, in the history database, the former number of transitions from $\mathcal{S}t_i$ to $\mathcal{S}t_j$ ($\mathrm{N}_{\mathcal{S}t_i\mathcal{S}t_j}$) and the number of all the possible transitions from $\mathcal{S}t_i$($\mathrm{N}_{\mathcal{S}t_i\mathcal{S}t_k}$) are already stored. Thisinformationis updated, and all the states and transition probability are represented and calculated accordingly.

The *prediction process* is mainly based on the results of the *classification phase*in order to predict the next user's intention and service. This prediction process is based on the semantic matching between the user's immediate intention and context and those of the user's behavior model states. Similar to the discovery mechanism, the semantic matching is based on ontologies in order to calculate the intentional and contextual matching scores. The final matching score represents the sum of the intention matching score and the context matching score. This information is stored with the state identifier. And going through all the states of the model, we can determine the state the most similar to the current user's situation.Subsequently, if a state is identified, then the next state is selected based on the transition probabilities. This transition probability must exceed a certain threshold. If several successor states are retrieved, then the one having the highest transition probability is chosen. By this choice, we can anticipate the user's future needs by offering him the most appropriate service that can interest him.

5. Evaluation

In order to evaluate the proposed service discovery and prediction mechanisms, we use the test collection OWLS-TC2[22] that we extend in order to include intentional and contextual information. Besides, we create a database containing user's traces, recognized clusters and the user's behavior model. We mixed a set of arbitrary traces with others following a scenario representing a well-defined user behavior. Thereafter, we launch the *clustering* algorithm on the set of traces in order to determine all the recognized clusters. Then, we executethe *classification*algorithm on all traces and clusters in order to update the user's behavior model stored in the database.

The service discovery and prediction mechanisms were implemented using Java language. Moreover, they are based on our OWL-SIC API (*OWL-S Intentional & Contextual*)[18], Jena[23] and the reasoner Pellet[24].

As part of our experiments, we deployed our algorithms on a machine Intel Core i5 1.3 GHz with 4 GB memory. The purpose of these experiments is to evaluate the validity of our algorithms and their feasibility. Thus, we formulate a set of user's requests relatives to the travel domain. These request are represented by the user's intention and his current context.These requests are formalized according to three different distributions. The *first distribution* considers requests that are very similar to the service (*service discovery*) or clusters centroid (*service prediction*). Then, the *second distribution*illustrates situations where the elements describing the intention and/or the context are not described in ontologies while there are services or clusters that are similar to this request. Finally, the *third distribution*shows the influence of the threshold by presenting in this distributionrequests that are within the limits of the threshold and others that are beyond the threshold.

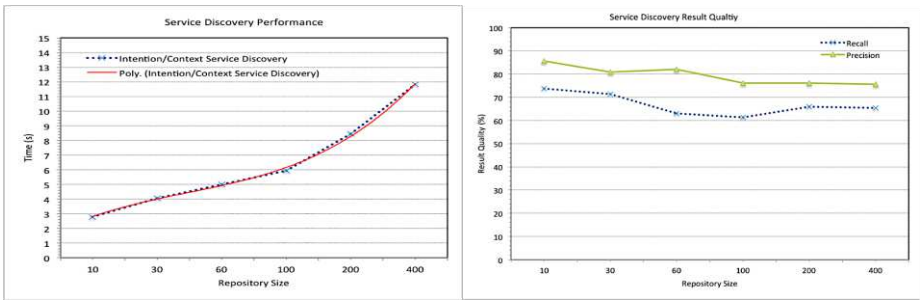

Fig. 4 a) the Service Discovery performance, b) The service discovery result quality

Our first experiments concern the *evaluation of our service discovery mechanism*. We measure the performance of the discovery algorithm by varying the number of services between 100 and 400. As illustrated in Fig. 4.a, the execution time follows a polynomial trend of degree three varying from 2,79 s for 10 services to 11,82 s for 400 services. However, even if this time still higher, we can observe that despite the fact that we have increased the number of services over forty times, the response time has only increased by for times.

Besides, in order to measure the quality of the result, we cover the two most useful quality metrics: *precision* and *recall*[15]. Through the experiments, we observe that the precision and recall are interesting factors when considering the intention and context in the service discovery. The result presented in Fig. 4.b shows that we obtained a higher precision percentage, about 80%. This indicates that our service discovery algorithm has a greater chance to retrieve

the most appropriate service according to user's intention and context. However, the good results of precision are accompanied by less interesting results concerning the recall, as illustrated in Fig. 4.b. We can observe that the average recall approximates the 67%. These results can be explained by the evaluation of some situations that can harm the results quality. For example, we have described some user's request where the elements of the intention are not described in ontologies, while it exists a set of services able to satisfy this intention in the current user's context.

Our second experiments concern the *evaluation of our service prediction mechanism*. We measure the performance of our algorithms with respect to the number of clusters, situations and states, by measuring the average processing time. For example, the execution time of the prediction algorithm is measured by varying the number of states in the user's behavior model, between 8 and 168 states. As illustrated in Fig. 5.b, the execution time follows a polynomial trend of degree three, like the service discovery algorithm, from 1,63 s for 8 states to 4,16 s for 168 states. We increased the number of states over twenty five times, while the execution time has only increased by two and half times. This allows us to validate the feasibility of our prediction algorithm.

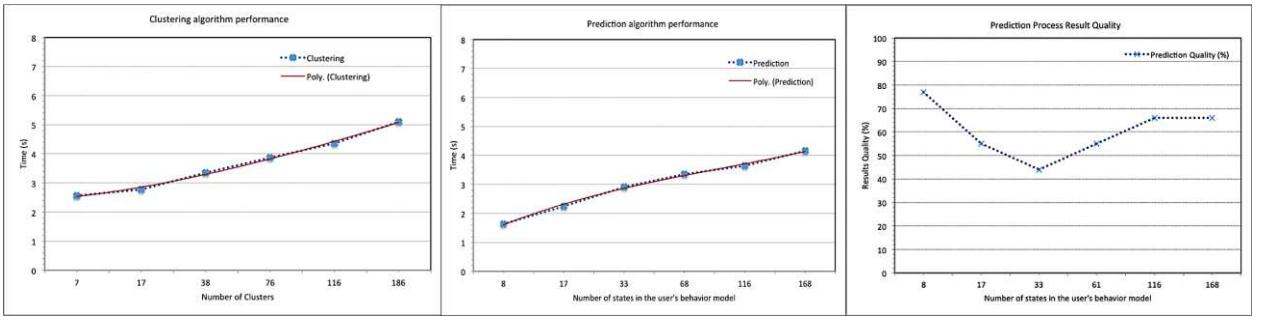

Fig. 5 a) The clustering performance, b) The prediction performance, c) The prediction process results quality

Besides, in order to measure the result quality, we use a *quality metric* inspired from the *precision* used to evaluate the service discovery. This metric is used to check whether the predicted service is the one that is expected or not.We determined previously the service thatthe prediction algorithm must predict for a given user's request, based on the user's behavior model. Then, we compared this service with the service returned by the algorithm.

We illustrate in Fig. 5.cthe quality percentage achieved by the algorithm by varying the number of states. This percentage represents the average quality obtained for all theuser's requests.The results presented in Fig. 5.c indicate that the prediction algorithm has a good quality that is around 60%. These results can be explained by the evaluation of certain situations that significantly degrade the results quality obtained. For example, in the case where situations are described by intentions where the verb and/or the target are fairly generic or specific, we obtain a quality in some cases below 45%. Thus, when the system designer sets very high threshold settings in the prediction algorithm, some clusters or states that can meet the immediate user's intention in his current context will not be selected, and this contributes to degradation of the results quality.

The analysis of these results shows the importance of the service discovery and prediction mechanisms. We believe that the proposed mechanisms allow really the selection of the service that fulfills the user's immediate needs and the anticipation of his future need. This is thanks to both its intentional approach, which is more transparent to the user, and its contextual approach that restricts services to those that are valid. However, it is important to note that we cannot get that good result if the system designer does not establishes from the beginning a rich description of the available services and the different ontologies and the most appropriate threshold setting.

6. Conclusion

In this paper, we have proposed a user-centric approach for service discovery and prediction considering PIS. This approach is needed to hide the complexity of these systems and to achieve the transparency required by their users. It enhances PIS transparency and proactivity throw service discovery and prediction mechanisms, defined considering the user's point of view. These allow us not only offering user the most suitable services given his current intention and context, but also to anticipate the future user's needs in a fairly understandable way.

By this approach, we believe contributing to the improvement of PIS transparency and proactivity through a user-centric perspective focusing on the intentions that services satisfy in a given context. Moreover, evaluating the user acceptation of the proposal requires applying it in a real case study. Such evaluation should consider the final user's point of view. It should consider the user acceptance, mainly considering the prediction mechanism,as well as the level of transparency perceived by these users.As a future work, we expect to evaluate our approach in a large-scale in order to validate its usefulness and compare it with the existing techniques.

References

1. Kourouthanassis P.E., Giaglis G.M. A Design Theory for Pervasive Information Systems. *3rd International Workshop on Ubiquitous ComputingIWUC* 2006; 62-70
2. Kaabi R.S., Souveyet C. Capturing intentional services with business process maps. *1st IEEE International Conference on Research Challenges in Information Science RCIS* 2007; 309-318
3. Dey A. Understanding and using context.*Personal and Ubiquitous Computing* 2001;5(1): 4-7
4. Chaari, T., Laforest, F., & Celentano, A. Adaptation in context-aware pervasive information systems: the SECAS project. *Journal of Pervasive Computing and Communications*, 2007: **3**(4), 400-425.
5. Toninelli, A., Corradi, A., & Montanari, R. Semantic-based discovery to support mobile context-aware service access. *Computer Communications* , 2008 : **31**(5), 935-949.
6. Vanrompay, Y., Kirsch-Pinheiro, M., Berbers, Y. Service Selection with Uncertain Context Information, In: Stephan Reiff-Marganiec and Marcel Tilly (Eds.), *Handbook of Research on Service-Oriented Systems and Non-Functional Properties: Future Directions*, IGI Global, 2011, p. 192-215.
7. Eikerling, H.-J., Mazzoleni, P., Plaza, P., Yankelevich, D., & Wallet, T. Services and mobility: the PLASTIC answer to the Beyond 3G challenge. White paper, Dec. 2007, http://plastic.paris-rocquencourt.inria.fr/promotion-material/white_paper_plastic_v1-3.pdf (Feb. 2014).
8. Mokhtar, S. B., Kaul, A., Georgantas, N., Issarny, Efficient Semantic Service Discovery in Pervasive Computing Environments. In: V. Steen, M. & Henning, M. (Eds.),*7th Int. Middleware Conference (Middleware'06)*, LNCS, Springer Berlin Heidelberg, 2006: **4290**, 240-259.
9. Fensel, D., Facca, F. M., Simperl, E., Toma, I. Semantic Web Services. Springer Berlin Heidelberg, 2011
10. Olsson, T., Bjurling, B., Chong, M., Ohlman, B. Goal Refinement for Automated Service Discovery. *3rd Int. Conf. on Advanced Service Computing*, 2011, 46–51
11. Santos, L. O. B. da Silva., da Silva, E. G., Pires, L. F., van Sinderen, M. Towards a Goal-Based Service Framework for Dynamic Service Discovery and Composition.3rd. Int. Conf. on Information Technology: New Generations, IEEE Computer Society, 2009, 302-307
12. Mayrhofer, R. An Architecture for Context Prediction. *PhD thesis*, Johannes Kepler University of Linz, 2004.
13. Sigg, S., Haseloff, S., David, K. An Alignment Approach for Context Prediction Tasks in UbiComp Environments,*IEEE Pervasive Computing*, 2010: **9**(4), 90-97.
14. Adomavicius, G., Tuzhilin, A. Context-aware recommender systems. In: Ricci, F., Rokach, L., Shapira, B., Kantor, P.B. (Eds.), *Recommender Systems Handbook*, Springer, 2011, p. 217-253.
15. Xiao, H., Zou, Y., Ng, J., Nigul, L. An Approach for Context-Aware Service Discovery and Recommendation,*IEEE International Conference on Web Services (ICWS)*, 2010, 163-170.
16. Vanrompay, Y., Berbers, Y. A Methodological Approach to Quality of Future Context for Proactive Smart Systems. In: Andreev, S.; Balandin, S. & Koucheryavy, Y. (Eds.),*Internet of Things, Smart Spaces, and Next Generation Networking*, LNCS, 2012:**7469**, 152-163
17. Baltrunas, L., Ricci, F. Experimental evaluation of context-dependent collaborative filtering using item splitting. *User Modeling and User-Adapted Interaction: Special issue on Context-Aware Recommender Systems*, 2013:24, 7-34.
18. Najar S, Kirsch-Pinheiro M, Souveyet C. Enriched Semantic Service Description for Service Discovery: Bringing Context to Intentional Services. *International Journal on Advances in Intelligent Systems*, 2012: **5**(1&2), 159-174
19. Paolucci M, Kawamura T, Payne T, Sycara K. Semantic Matching of Web Services Capabilities. In Horrocks I., Hendler J., editors. *The Semantic Web (ISWC 2002) - Lecture Notes in Computer Science*. 2342, 2002: 333-347, Springer.
20. Najar S, Kirsch Pinheiro M, Souveyet C, Steffenel A. Service Discovery Mechanism for an Intentional Pervasive Information System. *International Conference on Web Services ICWS*, 2012: Honolulu, United States, 520-527
21. FellerW. An Introduction to Probability Theory and its Applications. 1968, New Jersey: Wiley
22. OWLS-TC: http://www.semwebcentral.org/projects/owls-tc/
23. Jena Semantic Web Framework, http://jena.sourceforge.net/
24. Pellet, http://www.mindswap.org/2003/pellet/ index.shtml