Service Discovery Mechanism for an Intentional Pervasive Information System

Salma Najar, Manuele Kirsch Pinheiro, Carine Souveyet, Luiz Angelo Steffenel

Service Discovery Mechanism for an Intentional Pervasive Information System

Salma Najar, Manuele Kirsch Pinheiro, Carine Souveyet
Centre de Recherche en Informatique - Université Paris1
90 rue de Tolbiac, 75013 Paris – France
Salma.Najar@malix.univ-paris1.fr,
{Manuele.Kirsch-Pinheiro, Carine.Souveyet}@univ-paris1.fr

Luiz Angelo Steffenel
CReSTIC – Syscom Team
Université de Reims Champagne-Ardenne
51100 Reims - France
Luiz-Angelo.Steffenel@univ-reims.fr

***Abstract*—Pervasive Information System (PIS) provides a new vision of Information System available anytime and anywhere. The users of these systems must evolve in a space of services, in which several services are offered to him. However, PIS should enhance the transparency and efficiency of the system. We believe that a user-centric vision is needed to ensure a transparent access to the frequently changing space of services regardless of how to perform it. In this paper, we propose a new approach of PIS, both context-aware and intentional called IPIS. In this approach, services are proposed in order to satisfy user's intention in a given context. Then, we propose a context-aware intentional service discovery mechanism. Such mechanism is based on an extension of OWL-S taking into account the notion of context and intention. We present in this paper IPIS platform. Then, we detail the proposed service discovery mechanism and present experimental results that demonstrate the advantage of using our proposition.**

Keywords-component; pervasive information system; service orientation; intention; contexte-aware service;

I. INTRODUCTION

Nowadays, with the development of mobile technologies, we observe a shift in Information Systems. Instead of having Information Technology in the foreground, triggered and manipulated by users, we witness that IT gradually resides in the background, monitoring user's activities, processing this information and intervening when required [8]. In other terms, we are observing the emergence of a *Pervasive Information System* (PIS) that intends to increase user's productivity by making IS available anytime and anywhere.

Contrarily to traditional IS, whose interaction paradigm is the desktop, PIS deal with a multitude of heterogeneous devices, providing the interaction between the user and the physical environment [8]. We believe that PIS should evolve in a *space of services* offering to users a set of heterogeneous services helping them to accommodate their needs.

Weiser [20] suggests that pervasive environment will be characterized by its transparency and homogeneity. Twenty years later, we can notice that this pervasive environment meant to be an invisible or unobtrusive one, represents a technology-saturated environment combining several devices highly present and visible. PIS have to face such an environment, in which a rapidly evolving and increasing number of services are available, with multiple implementations. However, details concerning such implementation are still too complex for the user, who wants just a service that satisfies his needs. This complexity requires considerable effort from the user in order to understand what is happening around him and in order to select the service that best fulfills his needs.

In order to reach such vision of PIS, these systems should enhance its transparency. We believe that this will only be possible through a user-centric vision that ensures a transparent access to the space of services without any technical details concerning how to perform it in a given context. Thus, we propose a service discovery mechanism guided by user's intention and context in order to hide implementation complexity, and consequently achieving the transparency promised. First, the notion of intention can be seen as the goal that we want to achieve without saying how to perform it [7] or as a goal to be achieved by performing a process presented as a sequence of intentions and strategies to the target intention [3]. In other words, an intention represents a *requirement that a user wants to be satisfied without really care about how to perform it or what service allows him to do so*. Then, the context concept represents *any information that can be used to characterize the situation of an entity (a person, place, or object)* [4].

By adopting this vision, we propose to improve the transparency by considering, on the one hand, the intention service allows user to satisfy, and on the other hand, context on which this intention emerges. Thus, we advocate that the selection of the service that satisfies user's intention, in PIS, is valid in a given context, and a context influences the manner to satisfy user's intention and the execution of the service that support it. We believe this relation can be explored for service discovering purposes: by observing intention and context, we can obtain a more precise service discovery mechanism, offering most suitable services.

In this paper, we present an *intentional pervasive information system* platform (IPIS), which enhances the transparency of the system by proposing a service discovery mechanism guided by user's intention and context. The service discovery, based on these two concepts (context and intention), will help users by discovering for them the most appropriate service, without carrying about any technical details concerning how to perform it in a given context.

This paper is organized as follows: Section II presents an overview on related works. Section III introduces our IPIS platform, while section IV details our proposed service discovery process. The section V presents the implementation of the service discovery, while the section VI presents the experiments results. Finally, we conclude in the section VII.

978-0-7695-4752-7/12 $26.00 © 2012 IEEE
DOI 10.1109/ICWS.2012.84

II. RELATED WORK

In the last decade, an important change has been performed on IS. Those systems become, with the growth of information technology, accessible for users on many different circumstances and too complex for them. These changes lead to the notion of PIS. *Pervasive Information System* (PIS) constitute an emerging class of IS, in which information technology is gradually embedded in the physical environment, capable of accommodating user needs and wants when desired [8]. This author points out, as main characteristics of PIS, not only the heterogeneity of device types, but also the property of *context-awareness*, as a result of the artifacts capabilities to collect, process and manage environmental information. Therefore, when regarding the literature, one may observe two complementary tendencies proposing a new vision of the information system: context-aware approaches and intention-based approaches.

Toninelli et al. [19], consider that, in pervasive scenarios, users require context-aware services that are tailored to their needs, current position, execution environments, etc. These authors propose a personalized semantic-based service discovery that aims at integrating semantic data representation and matchmaking support with context management and context-based service filtering.

Similar to Toninelli *et al.* [19], Ben Mokhtar *et al.* [2] propose a context aware semantic matching of services. They propose, for service discovery purposes a language for semantic specification of functional and non-functional service properties named EASY-L and a corresponding set of conformance relations named EASY-M.

Xiao *et al.* [21] consider context-aware services and the dynamicity of pervasive environment. They use ontologies to enhance the meaning of a user's context values and automatically identify the relations among different context values. Based on these relations, they discover and select the potential services that the user might need.

These authors [2][19][21] emphasize the importance of the context on service discovery. They focus on the technical level that is still too complicate for users who would prefer just a service that satisfy their needs.

A different point of view is given by works such as [1] [9][12][17], which pointed out the importance of considering user's requirements on service orientation. Among these works, [7][17] propose a service oriented architecture based on an intentional perspective. Such architecture proposes the notion of *intentional service*, which represents a service focusing on the intention that it allows satisfying it rather than the functionality it performs. Such service represents an alternative for bridging the gap between low level, technical software-service descriptions and high level, strategic expressions of business needs for services.

Aljoumaa *et al.* [1] propose, based on the Intentional Services Model (ISM) proposed by [17], an ontological based solution to help matching user's needs formulated in business terms as goals with the intentions of services published in an extended registry.

Mirbel et al. [9] propose using semantic models in order to enrich needs description and to propose means of reasoning for intention based service discovery. This approach express user's requests using semantic Web technologies and help such users to find available services that fit these requests.

Olson *et al.* [12] believe that by using goals, services can be described on any arbitrary and useful level of abstraction. According to these authors, through a goal refinement algorithm, goals can be used not only for describing services, but also for improving the performance of service discovery.

None of these works considers the notion of context, contrary to *Bonino et al.* [3], which propose a goal-based dynamic service discovery framework that uses context information. This approach is centered on the use of context information for filtering the input of the user's request. Besides, These authors identify in advance, for each domain, a set of specific intentions and the different tasks allowing the fulfillment of them, which is quit restrictive since it cannot satisfy an intention that is not identified in advance.

All these works advocate for a more user-centric view of IS, by proposing either an intentional or a contextual approach. However, we believe that such user-centric vision can only be reached if one considers the closely relation between the notion of context and intention. We advocate that a service discovery mechanism based on user's context and intention is needed in order to provide answers to questions such as "*why a service is useful in a given context?"* or "*in which circumstances a service need raises?*". For us, *an intention is valid in a given context and a context has an influence on the satisfaction of this intention*. However, as far as we know, none of the previously works proposes a pertinent service discovery mechanism taking into account context and intention.

III. TOWARDS AN INTENTIONAL PERVASIVE INFORMATION SYSTEM

In this paper, we present a user-centric view of PIS called *Intentional Pervasive Information System* (IPIS). IPIS purpose is to propose to users the most appropriate service in a transparent way. It intends to enhance service discovery, on a space of services, according to the user's context and intention, without an explicit help from the user.

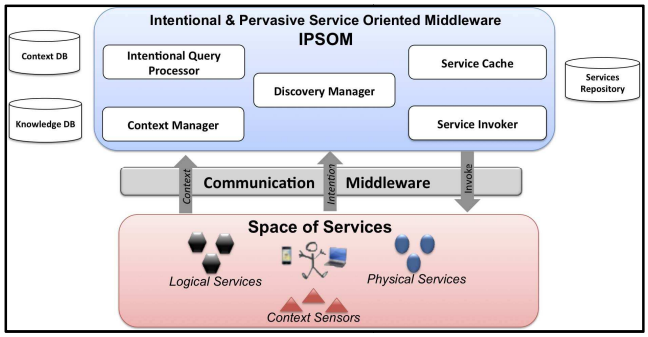

Figure 1. IPIS: Global Architecture

We advocate that understanding user's intention can lead to a better understanding of the real use of a service and consequently the selection of a suitable service that satisfies

user's needs. On the other hand, contextual information plays a central role since it influences the selection of the best strategies of the intention satisfaction.

We present in this section a global vision of IPIS platform as illustrated in Figure 1. IPIS is composed of three main layers: space of services (section III.A), communication layer (section III.B) and the Intentional Pervasive Service-Oriented Middleware (section III.C).

A. Space of services

The evolution of IS into PIS lead us to consider PIS as more than a set of logical services. Indeed, with the development of pervasive technologies, IT becomes embedded in the physical environment, offering innovative services to the users evolving on this environment. Contrarily to traditional IS, PIS may offer both logical and physical services. For us, in a *pervasive information system,* users evolves in a *space of services* offering them a set of heterogeneous services whose focus is to accommodate user's needs. Thus, we introduce the notion of *space of services* as a way to develop such user-centric view through a space that includes not only *logical services,* representing traditional information systems themselves, but also *physical services* embedded on the physical world. This space allows the representation of knowledge about users and their environment, in order to discover the most appropriate service that satisfies a user's intention in a given context.

B. Communication Layer

The communication layer aims at connecting the space of services with our *Intentional Pervasive Service-Oriented Middleware* 'IPSOM'. User requests, context capture, service invocation, etc. are sent to IPSOM through this layer.

C. IPSOM: Intentional Pervasive Service Oriented Middleware

The Intentional Pervasive Service oriented middleware 'IPSOM' represents the core of IPIS. Its main purpose is to satisfy user's intention by discovering and selecting the most suitable service in a given context. This middleware, presented in Figure 1 contains four main modules: *context manager, intentional query processor, service invoker and service discovery*.

1) Context Manager

The *context manager* purpose is to hide the context management complexity by providing a uniform way to access context information. It receives raw context data from different physical and logical sensors (GPS, RFID...) and interprets such data in order to derive context knowledge represented in a higher level. This knowledge is stored in a knowledge database using a context model.

Based on a previous study of context modeling requirements [11], we decided to rely on semantic modeling in order to represent context knowledge in an extensible context model. This context model is based on a multi-level ontology that provides a vocabulary for representing knowledge and describing context information. This multi-level ontology consists in an upper level, defining general context information, and a lower level, with specific context information. The advantage of using such multi-level ontology is its extensibility. This semantic modeling allows the knowledge sharing and its reuse. Besides, the use of ontology for formalizing context information enables the use of powerful logic reasoning mechanisms [11].

2) Intentional Query Processor

The *Intentional Query processor* (IQP) is in charge of processing user's request. Such request represents user's intention, expressed by a *verb*, a *target* and a set of optional *parameters,* according to a specific template [7][17]. The IQP enriches this request with context obtained from context model. This enriched request, represented in XML format, is then transferred to the discovery module.

3) Service Invoker

When a service is discovered and selected by the *discovery manager*, the URI of this selected service is sent to the *service invoker*, which is in charge of invoking and executing it.

4) Discovery Manager

Discovery manager (DM) is in charge of discovering most suitable services, according to the user's context and intention. This process is based on a semantic description of services and on a matching algorithm detailed in section IV.

The service discovery process is launched when the IQP sends the enriched request to the DM. Then, DM loads the semantic description of the available services and launches the matching process on all the available services.

IV. Context-Aware Intentional Service Discovery

In order to enhance the transparency and efficiency of PIS, we propose a new mechanism for service discovery based on our service description enriched with intentional and contextual information [10].

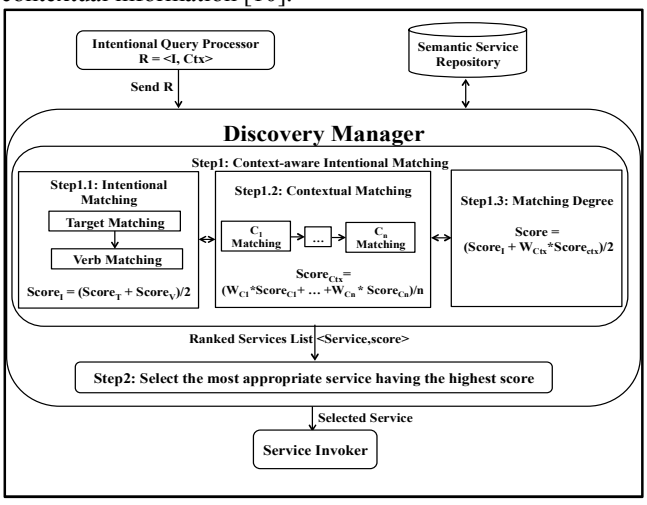

Figure 2. The context-aware intentional service discovery process

The intention concept is used to expose services and to implement a user centric vision of the PIS in a given context. We propose a context-aware intentional service discovery based on a semantic matching algorithm. This matching is a two-steps process illustrated in Figure 2.

First of all, we match the user's intention with the intention that the service satisfies (step 1.1). Second, we match the services context conditions with the user's current context (step 1.2). Finally, we calculate the matching degree between the user's request and the provided service, and we include the service with its obtained score in a list (step 1.3). These steps, detailed in section IV.B, are done for all the available services. Then, from the resulted list, we select the service having the highest score (step 2).

A. Enriched service description

The service description used in the service discovery process is based on our extension of OWL-S, which includes the information concerning both context and intention that characterize a service [10].

Actually, we consider that a user requires a service because he has an intention that the service is supposed to satisfy. Such intention emerges on a given context, which also characterize the service. We enriched OWL-S service description with both intention and context description [10]. Intention is described by adding a new sub-ontology, which refers the intention that a service can satisfy. Context is described by a context element pointing out an external file, which allows service provider to easily update context information related to the service description itself. More details about our extension of OWL-S are presented in [10].

B. Context-aware intentional service discovery algorithm

The service discovery algorithm that we propose in this paper performs a semantic matching process in order to offers the most appropriate service to the user. Once all the available services have been loaded, the matching algorithm may proceed. The goal of this matching algorithm is to rank the available services based on their contextual and intentional information and select the most suitable one for the user. This algorithm compares semantically user's intention with the intention that the service satisfies and user's current context with the service's context conditions.

Algorithm 1 Service Discovery

```
1:  Procedure SERVICEDISCOVERY (C_U, I_U, S)
2:  S_ranked = ø
3:  I^score = 0
4:  C^score = 0
5:  S^score = 0
6:  for ∀s ∈ S do
7:      I^score = IntentionMatching(I_U, I_S)
8:      if I^score > k then
9:          C^score = ContextMatching (C_U, C_S)
10:         S^score = (I^score + (w_i * C^score))/2
11:     end if
12: if S^score > l then
13:     S_ranked.add (s, S^score)
14: end if
15: end for
16: return S_ranked
17: end procedure
```

Figure 3. The context-aware intentional service discovery algorithm

The Figure 3 presents the proposed discovery algorithm. For all the available services, we calculate the matching score S^{score} between the user's request and the service (line 6-11). First, we calculate the intention matching score I^{score} between the user's intention I_U and the service intention I_S (line 7). As we mentioned above, the intention expresses the user's requirement that he wants to be satisfied by the system. It is composed of two mandatory elements: *verb* (V) and *target* (T). The *verb* exposes the action allowing the realization of the intention. Then, the *target* represents either the *object* existing before the satisfaction of the intention or the *result* created by the action allowing the realization of the verb [7][17]. Thus, to define the matching degree between user's intention I_U and service's intention I_S, we calculate: (1) the matching degree between user and service targets (respectively T_U and T_S); (2) the matching degree between user and service verbs (V_U and V_S), and finally (3) the intention score representing the sum of the verb and target matching score. Once intention matching is performed, we proceed with context match, in which we calculate score C^{score} from matching between user's current context and contextual conditions established by service description (line 9). Both scores (I^{score} and C^{score}) are then used to calculate service final score S^{score} (line10). Both matching are detailed in next sections.

1) Intention matching

The intention matching is based on the use of *ontologies*, *semantic matching* and *degree of similarity*. Besides, regarding to the intention formulation, the intention matching is especially based on a *verb* and *target* matching. According to this, we use an *ontology of verb* and a *domain-specific ontology* representing the possible targets in a specific domain. The degree of similarity represents a distance calculated based on the semantic link between two concepts in the ontologies. Thus, the intention matching is calculated based on two relations, *TargetMatch* and *VerbMatch*, used to define the *IntentionMatch* between user's intention $I_U = <V_u, T_u>$ and service's intention $I_S = <V_S, T_S>$ as follows:

$$IntentionMatch(I_U, I_S) = \begin{array}{c} \forall V_U, \exists V_S : VerbMatch(V_U, V_S) \\ And \\ \forall T_U, \exists T_S : Target Match(T_U, T_S) \end{array}$$

The target matching relation compares concepts defined in a domain-specific ontology, from required targets T_U and provided target T_S. The evaluation of how well a required target matches a provided target is typically based on the subsumption hierarchy used to find which a provided concept can fulfill a requested concept. In order to perform such matching, our algorithm is based on the matching algorithm proposed by [15], using the following 4 levels:

- ***Exact***: the required concept is equivalent to the provided concept
- ***Plug-In***: the provided concept subsumes the required concept
- ***Subsume***: the required concept subsumes the provided concept
- ***Fail***: there is no subsumption between the two concepts.

Thus, the target matching score is calculated based on the function *getScoreTarget* that takes as input the domain-specific ontology, the user's target and the service's target. The output of this function represents the degree of similarity between these targets calculated based on the distance between them in the domain-specific ontology. The score of the target matching is calculated as illustrates TABLE I.

TABLE I. TARGET MATCHING SCORE

Matching relation	Distance	Score
Exact	0	**1**
Fail	-1	**0**
Plug-in / Subsume	d	**1 / (d+1)**

Then, if the target score is greater than a fixed threshold we start the verb matching step, else we can conclude that the corresponding intentions are not similar and consequently the corresponding service can not satisfy the user's intention.

The verb matching relation is based on a verb ontology, which contains a domain-specific set of verbs, their different meanings and relations. Each verb relation associates a verb with related general verbs, more specific verbs and verbs sharing a common meaning. Thus, we propose to categorize verb matches according to five level of matching:

- ***Exact*:** the required verb is equivalent to the provided verb;
- ***Synonym*:** the required verb shares a common meaning with the provided verb;
- ***Hyponym*:** relationship of subordination between the required verb and the provided verb with a more general sense;
- ***Hypernym*:** relationship of subordination between the required verb and the provided verb with a more specific sense;
- ***Fail***: there is no relation between the verbs.

Such levels are based on a relation *Property (P, C_1, C_2)*, in which C_1 is a required concept, C_2 is a provided concept and P is the related property between C_1 and C_2. For example, *Reserve.hasSynonym = Book*. This relation is defined as follows:

$$\boxed{\mathrm{Property}(P, C_1, C_2) = \forall C_1, C_2, \exists P : C_1.P = C_2}$$

Thus, the verb matching score is calculated based on the function *getScoreVerb* that takes as input the verb ontology, the user's verb and the service's verb. The output of this function represents the score of the verb matching degree. This score is calculated based on the relation between the two verbs. First, this function determines the relation property between them. Then, the score of the verb matching is calculated as illustrates TABLE II.

TABLE II. VERB MATCHING SCORE

Relation Property	Score
Exact	**1**
Synonym	**0,9**
Hyponym	**0,7**
Hypernym	**0,5**
Fail	**0**

The final score of the intention matching is represented as the sum of the verb matching score and the target matching score. Thus, if the intention matching score is greater than a given threshold, the second step representing the contextual matching is triggered (line 9).

2) Context matching

The context description for a user (C_U) or a service (C_S) represents a set of observable context elements, in which $C_U=\{c_j\}_{j>0}$ and $C_S= \{c_i\}_{i>0}$. Each context element is described by an *entity* (to which the context element refers), an *attribute* (that characterizes the property that we observe) and a set of observed *values*. In order to define the relation *ContextMatch*, we consider the relation *ContextElementMatch* that matches individually the different context elements constituting the user's and service context descriptions C_U and C_S. This relation is presented as follows:

$$\boxed{\mathit{ContextMatch}(C_S, C_U) = \forall c_i \in C_S, \exists c_j \in C_U : \mathit{ContextElementMatch}(c_i, c_j)}$$

The context element match proceeds as follow: for each c_i and c_j, we (i) match the c_i.entity with the c_j.entity; if the matching score between them is higher than a given threshold then we (ii) match the c_i.attribut with the c_j.attribut; if the matching score between them is higher than a given threshold then we (iii) match the different values one by one.

The observable context elements can be divided into several types. Thus, the context might be represented as a multi-dimensional space. Context information values are distinguished between numerical or non-numerical types. In order to take into account this diversity and the multi-dimensional space representing the context, the relation *ContextElementMatch* identifies the nature of the context element value and accordingly triggers the suitable matching function 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, lower-than).

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 description, if the captured value of the user's device bandwidth is really higher than 12500, then we return an exact match. Thus, the context matching score C^{score} (line 9) is calculated as the sum of the scores of each context element, and represented as follows:

$$\boxed{C^{score} = \sum_{i=1}^{n} w_i f(type, c_i, c_j)}$$

The *weight* 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.

V. SERVICE DISCOVERY MECHANISM

A. Overview

Service discovery algorithm proposed in section IV was implemented using Java language, based on a basic service discovery mechanism [18]. Such basic mechanism is organized around three main interfaces: ServiceManager, PersistenceManager and SearchEngine. The ServiceManager represents the entry point to the application and offers a set of methods allowing ontology management and service discovery and selection. PersistenceFacade interface acts as a façade between the PersistenceManager and the database to access service and ontology descriptions, handling ontologies and service descriptions. SearchEngine is in charge of searching an appropriate service for a given request. It uses a MatcherFacade interface that acts as a facade between the SearchEngine and the API to operate

service descriptions. This *MatcherFacade* allow easily extend discovery mechanism by adding new discovery algorithms. Next sections detail the extensions we made on this basic service discovery mechanism.

B. Service Description Implementation

The context aware intentional service discovery is based on an extension of OWL-S that we presented in a previous work [10] This extension includes the intention that a service satisfy and context conditions in which a service is valid. Thus, based on the OWL-S API Mindswap [13], we develop in Java the OWL-SIC API, which implements the service description according to his intention and context. In order to evaluate the proposed implementation, the service retrieval test collection OWLS-TC2 [14] is used. We chose this test collection because it provides a large number of services from several domains, test queries and relevant ontologies, in spite of containing only basic service descriptions based on OWL-S. The collection is intended to support the evaluation of the performance of OWL-S service matching algorithms.

We extend service descriptions proposed by OWLS-TC2 in order to include on those services intentional and contextual description. These service descriptions are used for evaluating proposed context-aware intentional service discovery mechanism in order to search the most interesting service according to user request and context.

C. Service Discovery Implementation

In our architecture, the *ServiceManager* interface is in charge of supplying the search methods for client applications. Every search method shall propose a list of suitable services with their matching scores and then return the best-ranked service. It decides, based on the calculated scores, the service that should fits the request. This search service is based on a *Matchmaker* interface representing a service matching algorithm. In our implementation, the matchmaker is easily replaceable in order to support multiple discovery processes. The selection of what matchmaker should be launched through a simple properties file.

Two main implementations of matchmaker have been proposed. First, a reference-matching algorithm called *BasicMatchmaker* [18], based on to the input and the output information given by the user. The *BasicMatchmaker* searches and selects the services according to this information [18]. As a second implementation, we propose the context-aware intentional service matching algorithm presented in this paper. This matcher, called *ContextIntentionMatchmaker,* uses OWL-S extended API [10], Jena [6] and the reasoner Pellet [16]. It calculates, for each service in the knowledge base, a score based on the intention and the context from the user's request and from the service description. This matchmaker is based on two classes *ContextMatching* and *IntentionMatching*, which are in charge of calculating respectively the context and the intention scores. By separating score computation, it is possible to easily disable one of these classes in order to evaluate separately the impact of context or intention matching on the core. In order to evaluate the validity of our context-aware service discovery algorithm, we compare the results of these two matchmakers. The experimental results of such evaluation are presented in the next section.

VI. EVALUATION

As mentioned earlier in this paper, we generate a semantic repository containing a set of extended service descriptions based on the extended OWLS-TC2. Among the provided domains, we choose to evaluate our proposed service discovery process in the Travel domain. It represents about 200 service descriptions that we enriched with intentional and contextual information. The evaluation has been performed in Grid'5000 platform [5], which contains a set of heterogeneous clusters, as indicated in TABLE III

TABLE III. GRID'5000 CLUSTERS USED IN THE EXPERIMENTS

Cluster	CPU	Processor	Cores	RAM
Stremi (2011)	2	AMD 1.7 GHz	12	48
Graphene (2010)	1	Intel 2.53 GHz	4	16
Griffon (2009)	2	Intel 2.5 GHz	4	16
Capricorne (2006)	2	AMD 2 GHz	1	2
Bordeplage (2007)	2	Intel 3 GHz	1	2
Bordereau (2007)	2	AMD 2.6 GHz	2	4
Borderline (2007)	4	AMD 2.6 GHz	2	32

The purpose of our experiments is to evaluate the validity of our algorithm and the feasibility of our extended service descriptor. Two main observations emerge from this experiment: (i) *Scalability*: Whether the processing time is reasonable; (ii) *Result quality*: whether the algorithm can effectively select the most appropriate services.

A. Scalability

We measure the scalability of our service discovery algorithm with respect to the number of services and the capabilities of the nodes from different clusters, by measuring the average processing time. The Figure 4 illustrates the experimental results of processing time consumed when running our service discovery algorithm.

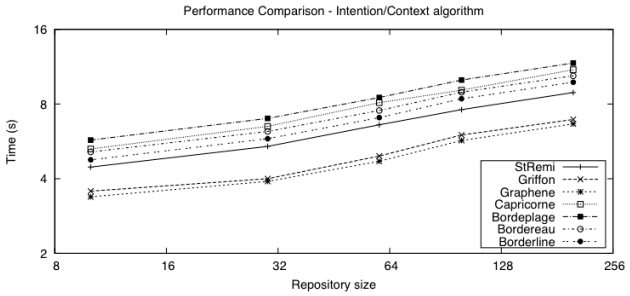

Figure 4. Intention/Context service discovery performance

The results demonstrate that the response time of the service discovery algorithm follows a logarithmic trend line, which allows confirming that the implemented mechanism has a good scalability. Then, we can notice that the use of 200 tested services only for the evaluation, which can be not

significantly large, stills sufficient to demonstrate the logarithmic behavior of our algorithm. Besides, we notice the greater impact of the processor family and the generation of the clusters. Nevertheless, we believe that these performances can be improved by parallelizing tasks on multi-core architectures.

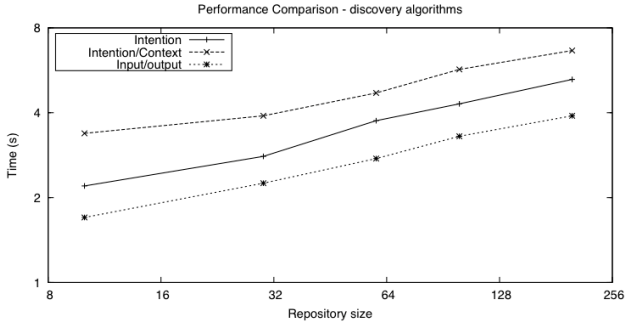

Figure 5. Service discovery performance comparison

The Figure 5 shows a comparison between three types of service discovery: (i) an Input/Output service discovery (*BasicMatchmaker*), (ii) an Intentional service discovery (*IntentionMatchmaker* with the context matching score disabled) and (iii) the Context-aware Intentional service discovery. For these three algorithms named, respectively, ***IO***, ***I*** and ***IC***, we measured response time using the same set of services. For the matter of clarity, we present the comparison of these 3 algorithms on the *Graphene* cluster. This cluster was chosen because its characteristics are closest to current off-the-shelf desktop/workstation, contrarily to the other clusters that are tailored for an HPC environment or have obsolete configurations.

The graph illustrates that the IC algorithm takes longer to process services. However, this difference on execution time does not represent a serious time difference from a user perspective. Actually, despite the fact that the input/output service discovery goes faster for selecting services, the response time of our algorithm still represents a reasonable processing time for selecting the most appropriate service.

This demonstrates the feasibility of our proposition on machines usually used as service repository, despite the fact that the implementation is not optimized to multi-cores architecture. Furthermore, we must consider that the IO algorithm uses a very simplified mechanism having a low cost but, as demonstrates the next section, presents a result quality considerably lower than the IC algorithm.

B. Result Quality

In order to measure the quality of the result, we cover the two most useful quality metrics: *precision* and *recall*. These two measures are defined in terms of a set of retrieved item and a set of relevant items. The *precision* represents how well a system retrieves only the relevant services, while the *recall* measures the ability of a system to retrieve all the relevant service [21]. The definition of recall and precision measures are defined as follows:

$$\textbf{Precision} = \frac{|\{\textbf{relevant items}\} \cap \{\textbf{retrived items}\}|}{|\{\textbf{retrieved items}\}|} \qquad \textbf{Recall} = \frac{|\{\textbf{relevant items}\} \cap \{\textbf{retrived items}\}|}{|\{\textbf{relevant items}\}|}$$

In order to evaluate these two measures, we formulate five user's requests relatives to the travel domain. These request are represented by the user's intention and his current context, as illustrates TABLE IV. In this scenario, the user is looking for surfing or hiking sport. Thus, he searches a destination where he can practice such sports. Then he wants to reserves a hotel or a Bed-and-Breakfast for the period. As a first step, we choose to evaluate our proposed service discovery algorithm using a simple scenario taking into account different or similar intentions in different contexts. In fact, this scenario represents a simple one, but it is sufficient to illustrate our approach.

TABLE IV. USER'S INTENTION IN A GIVEN CONTEXT

Intention	Context
Reserve Hotel	- Age >=18
Reserve BedAndBreakfast	- Age >= 18 - Season = Summer
Locate Sport Destination	-Age >= 18 - Season = Summer - City = Germany
Search Surfing Destination	-Age >= 18 -Surfing-Level=Beginner -Season=Summer -Weather=notDisturbed
Search Hiking	-Age >= 18 - Hiking Level=Confirmed -Weather=not disturbed -Health = Good

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 Figure 6 shows that we obtained the highest precision percentage compared to IO algorithm. We obtain about 99% of precision, while the IO algorithm obtains about 50% of precision. This indicates that our service discovery algorithm has a greater chance to retrieve the most appropriate service according to user's intention and context.

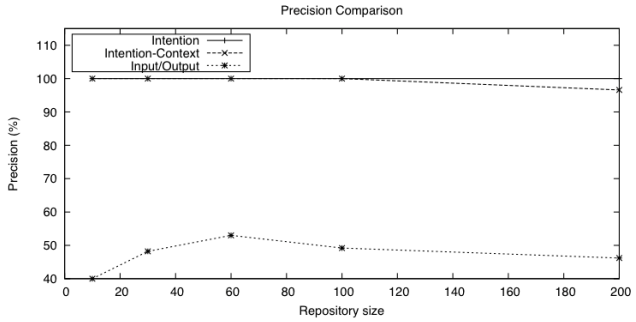

Figure 6. Precision Results

Besides, the results illustrated in Figure 7 show that we obtain an interesting recall about 95% compared to the IO algorithm, which reaches about 93.2%. However, the results obtained by the IO algorithm are circumstantial since this algorithm is not able to select a service adapted neither to user's context nor to his intention. In fact, the IO algorithm can only return all the services related to the request with a high rate of "false-positive" (indicated by its *precision*). This demonstrates that our IC algorithm is able to find all or almost services that can fulfill user's intention in a given context, with the lowest rate of "false-positive". These properties are parts of parameters, which contribute in the quality of the user's experience.

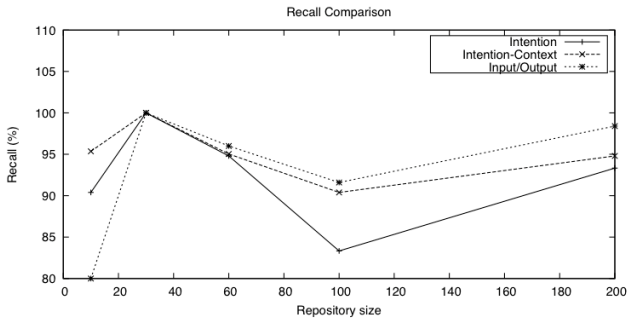

Figure 7. Recall Results

The analysis of these results, illustrated in Figure 6 and Figure 7, demonstrates that our proposition presents a more interesting result with a higher quality. We believe that our service discovery mechanism allows selecting the most appropriate service. And that is due to both its intentional approach, which is more transparent to users, and contextual approach that limits services to those that are valid.

VII. CONCLUSION

With the development of pervasive technologies, information systems become pervasive (PIS). Unfortunately, PIS provides several implementations of services that still too complex for user, who just requests a service satisfying his need. In order to enhance the transparency, we should select to the user the service that satisfies his need, without having to learn more details about the implementation or the constraints of used devices.

In this paper, we propose a user-centric vision of PIS that considers both user's intention and context. This vision allows us to obtain a more precise service discovery offering the most suitable service for the user. Thus, we implement a service discovery mechanism guided by user's intention and context. This service discovery mechanism is evaluated according to two aspects. The first one is the *scalability,* which allows analysing the feasibility of the proposed mechanism especially with respect to the growing of the service repository. The second one is the *result quality* based on two measures in services field called *precision* and *recall*. These two measures are used to analyse whether our mechanism really reached its goal. The experiments demonstrate that our algorithm is able to find, in a reasonable time, all or almost services that can fulfill user's intention in a given context, with the lowest rate of "false-positive".

Future works will be concentrated on the improvement of our experiments results. We expect to evaluate our service discovery mechanism in a more interesting real world scenario. Besides, these experiments will be tested on a more important number of services. Moreover, we opt to improve the implementation of the service discovery algorithm in order to reduce the processing response time. As a next step, our efforts will be focused also on the service prediction mechanism. Given the large amount of existing services and user's needs, our purpose is to help users by offering them, without their demand, personalized services that can interest them.

ACKNOWLEDGEMENTS

Experiments presented in this paper were carried out on Grid'5000 experimental testbed (*https://www.grid5000.fr*).

REFERENCES

[1] K. Aljoumaa, S. Assar, and C. Souveyet, "Reformulating User's Queries for Intentional Services Discovery Using an Ontology-Based Approach," 4th IFIP Int. Conf on New Technologies, Mobility and Security (NTMS), Paris, France, pp. 1-4, 2011.

[2] S. Ben Mokhtar, D. Preuveneers, N. Georgantas, V. Issarny, and Y. Berbers, "EASY: Efficient semAntic Service discoverY in pervasive computing environments with QoS and context support," Journal of Systems and Software 81(5), pp.785-808, 2008.

[3] L.O. Bonino da Silva Santos, G. Guizzardi, L.F. Pires, and M. Van Sinderen, "From User Goals to Service Discovery and Composition," ER Workshops, pp. 265-274, 2009.

[4] A. Dey, "Understanding and using context," Personal and Ubiquitous Computing, Vol. 5 n°1, pp. 4-7, 2001.

[5] Grid5000, https://www.grid5000.fr/ , 2011.

[6] Jena Semantic Web Framework, http://jena.sourceforge.net/.

[7] R.S. Kaabi, and C. Souveyet, "Capturing intentional services with business process maps," RCIS, pp. 309-318, 2007.

[8] P.E. Kouruthanassis, and G.M. Giaglis, "A design theory for pervasive systems," IWUC, pp. 62-70, 2006.

[9] I. Mirbel, and P. Crescenzo, "From end-user's requirements to Web services retrieval: a semantic and intention-driven approach," J.-H. Morin, J. Ralyte, M. Snene, "Exploring service science," First Int Conf, IESS, Springer, pp. 30-44, 2010.

[10] S. Najar, M. Kirsch-Pinheiro, and C. Souveyet, "The influence of context on intentional service," 5th Int. IEEE Workshop on Requirements Engineerings for Services (REFS'11) - IEEE Conference on Computers, Software, and Applications (COMPSAC'11), Munich, Germany, pp. 470 – 475, 2011.

[11] S. Najar, O. Saidani, M. Kirsch-Pinheiro, C. Souveyet, and S. Nurcan, "Semantic representation of context models: a framework for analyzing and understanding," J. M. Gomez-Perez, P. Haase, M. Tilly, and P. Warren (Eds), 1st Workshop on Context, information and ontologies CIAO, European Semantic Web Conference ESWC'2009, ACM, pp. 1-10, 2009

[12] T. Olsson, M.Y. Chong, B. Bjurling, and B. Ohlman, "Goal Refinement for Automated Service Discovery," 3rd Int. Conf on Advanced Service Computing, Service Computation, Rome, Italy, pp. 46-51, 2011.

[13] OWL-S API: http://www.mindswap.org/2004/owl-s/api/.

[14] OWLS-TC: http://www.semwebcentral.org/projects/owls-tc/ .

[15] M. Paolucci, T. Kawmura, T. Payne, and K. Sycara," Semantic matching of web services capabilities," in: Proceedings of the First International Semantic Web Conference, Springer LNCS 2342, Sardinia, Italy, 2002.

[16] Pellet, http://www.mindswap.org/2003/pellet/ index.shtml.

[17] C. Rolland, M. Kirsch-Pinheiro, C. Souveyet, "An Intentional Approach to Service Engineering," IEEE Transactions on Service Computing, Vol.3 n°4, pp. 292-305, 2010.

[18] S. Schulthess, "Construction of a registry for searching web service," Master Thesis, EFREI, Engineering School Paris, 2011.

[19] A. Toninelli, A. Corradi, and R. Montanari, "Semantic-based discovery to support mobile context-aware service access," Computer Communications, vol.31 n° 5, pp. 935-949, 2008.

[20] M. Weiser, "The computer for the 21st Century, " Sci Am 265(3), pp. 94–104, 1991.

[21] H. Xiao, Y. Zou, J. Ng, and L. Nigul, "An Approach for Context-aware Service Discovery and Recommendation," IEEE Int. Conf on Web Services (ICWS), Miami, pp.163-170, 2010.