Available online at www.elixirpublishers.com (Elixir International Journal)

Computer Science and Engineering

Elixir Comp. Sci. & Engg. 91 (2016) 38468-38472

ISSN: 2229-712X

Designing web services communities with UML oriented agents

Hela Limam and Jalel Akaichi
Department of Computer Science, ISG University of Tunis Bouchoucha, Bardo Tunisia.

ARTICLE INFO

Article history:
Received: 18 December 2012;
Received in revised form:
15 February 2016;
Accepted: 22 February 2016;

Keywords
Web Services,
Communities,
M-UML,
Mediator,
Health care system

ABSTRACT

The incredible growth of the information space, the hard competition between enterprises populating it, the need to collaborate and to share knowledge guiding to better decisions are factors that push Web Services providers to join each other into Web Services communities. Solutions proposed to tackle communities' management lack of a formal and clear design which may alter the system evolving and maintenance. For these reasons, we propose to design and to implement a system able to manage Web Services communities using an agent oriented approach. As illustration, we design and implement a case tool for managing e-health care Web Services communities.

Introduction

With the advance of the World Wide Web, Web portals related to business or leisure were created (O'Murchu et al., 2004). Many of them have proven to be highly popular and successful by acquiring millions of members. They offer tremendous opportunities for empowering users and organizations in various application domains including electronic commerce, travel, intelligence information gathering and analysis, health care, digital government, etc.

However, the technology to organize, search, integrate, and evolve these portals has not kept pace with the rapid growth of the available information space. Enjoying the above benefits using current Web technologies is a very complex process for end users.users and to enhance acting in groups according to common interests.

In fact Web Services communities introduce management problem resulting from the lack of a generic tool for community building and the absence of interoperability among different community support platforms. Community members often want to query, monitor, and discover information about various entities and relationships not only in their communities but also in others communities.

A critical challenge therefore is to design a system able to manage Communities taking into account many tasks such as discovering and updating communities then building relationships between them etc. Our solution is to design and to implement a system able to manage Web Services communities while addressing all of these issues.

This paper is organised as follows. In Section 2 we first Present the related work then we introduce our modelling tool in section3. The system requirements are detailed in section4. Our system architecture is exposed in section 5. Then, in Section 6, we illustrate our work a health care example for communities building. We also use the same example to show the communities relationships in section 7 and the query processing in section 8. Finally, section 9 presents our conclusions and future work.

Tele:
E-mail address: hela1.limam@laposte.net

Related Works

Two major areas of related research are building Catalog Service Communities and the use of a multi agent approach for managing Web Services Communities.

The project WS-CatalogNet (Benatallah et al., 2004) proposes a framework in which Catalog Service Communities are built, linked for interaction, and constantly monitored and adapted over time. This enabled a potentially large number of catalogs to act as one catalog to serve customers' queries. The approach is based on a hybrid of P2P data management paradigm and Web services architecture. It uses the notion of Catalog Services communities where catalogs catering for similar customer needs are grouped together, and form a single community (ie., a peer node). WS-CatalogNet consists of a set of integrated tools that allow for creating communities, registering e-catalog members, querying individual communities and routing queries among communities.

WS-CatalogNet work, so far, has focused on creating, linking and querying the community network. An evolution of the platform was proposed later. It consists investigating the ability to evolve from the initial design of the system. The evolution may involve capabilities like "discovering" members (ie, catalogs) or other communities (ie, peers), splitting a community, re-forming the relationships, etc.

In the agent context, (Yamada et al., 2004) discuss the multi agent-based simulation approach to analyze online community activities, and the design problem of the decision-making model of an agent that constructs multi-agent systems. And (Lacher & Koch 2000) outlined the possibilities communities provide for enhanced information management and filtering. For tackling the problem of interoperability among community support systems and they introduced an agent-based architecture for building community support systems and presented two projects that are built based on this architecture.

THE MODELLING TOOL

In the design of a communities' management system, the use of modeling tools has a decisive impact on the success of the system implementation. The use of UML seems to be crucial and beneficial since it has a well-defined syntax and semantics. However ,as our system is characterized by the mobility aspect especially while collaborative query processing between communities , the use of UML becomes inappropriate since it can not provide Mobility description for all views and aspects of the system. However, recently an extension to UML was proposed (Kassem et al., 2003) . It is called M –UML and it allows the explicit description of the mobility aspect .M –UML provides mobility description for all views and aspects of systems, hence covering all UML diagrams. It is a mobile agents modeling tool.

Similar to E-catalog (as Web Service), mobile agent can encapsulate business or application logic. Rather, mobile agents can dynamically, combine and execute such processes, and further offer multiple services or behaviors that can be processed concurrently. In order to move from system to another, or even to communicate with each other, mobile agents currently need a common platform on which they operate. Thus, they are useful for business partners only if theses actually share common platform. The consistent use of web service standards for description of capabilities, communication, and agent discovery would establish interoperability not only between different agent platforms but also between agent platforms and traditional web services. Thus, the advantages of two worlds can be combined.

In our system, software agents are not used for services communication front ends or as proxies. Rather, they are basic entities that encapsulate Web services.

REQUIREMENTS

Use Cases

The communities' management system aims to meet the requirements of a virtual collaboration system with both autonomous and collaborative Web services, by supporting communities whose members interact and form groups based on their common interests. The main requirements of a communities' management system can be resumed by satisfying the users' requests, caching the queries results for a future reuse and maintaining cache to remain available, these requirements are detailed in the following:

Creating and Updating Communities: The process starts with the elicitation of user requirements. In this phase, requirements about important and interesting topics in the domain are collected, the information goals of potential users of the portal are elicited, and preferences or expectations concerning the structure and layout of presented information are documented. Results of this very first phase constitute the input for the design of the communities' web portals.

- Building peer relationships between communities: Through a peer relationship between communities, they forward queries to each other. To form a peer relationship, community providers need to discover other communities whose domains are relevant/similar to their communities.
- Requesting a Service: The views and queries are described and formalized during the query development step. At first, their functionality is tested independently of the web site design. To express the information needs formally, the developer has to access the ontology, whereby additional rules or relations that define new views or ease the definition of queries may become necessary
- The query processing between communities: The collaborative query processing technique consists of two steps. Whenever a query is submitted to a community it does the following:

First Step: Identify the combination of members whose query capabilities, when put together, satisfy all constraints expressed in the query. The members can be local (belonging to the community), or external (belonging to the community peers).

Second Step: Send the sub-queries to the identified members and collect the results.

Queries answers storage: We develop a semantic cache to store queries answers. This semantic cache collects the answers after the query resolution .Its goal is the reuse of these results for similar requests and to store the links to the communities requested.

Actors

Even if we are using a mobile modeling tool, it may not always be the best solution to represent all the actors as agents. In function of the intelligence, autonomy and mobility level of each actor, one must carefully decide if it should be represented as a mobile actor. We do not propose a general rule for determining the chosen representation of an entity, but the designer's common sense and experience should make this choice adequate. We will consider bellow each actor and justify the chosen representation.

- The Community Manger: It is a mobile actor located at the community supports the following tasks: The community creation, the member (user/provider) subscription, the communities update, Accept service requests and matching a service request to potential provider(s).
- The Member Manager: It represents the members of the community. It assists members in performing the following tasks: request for member subscription and Interfaces human Community members.
- The Query Manager: This is a mobile actor that has the following characteristics: the capacity to migrate to other communities and to the potential provider's site, the capacity of collecting answers from other communities.
- The Web Services Provider: It is a mobile actor that performs the following tasks: search to discover communities of interests and join them via the registration process and it has the capacity to join or leave a community of interest at any time
- The User: This type refers to the users who have not yet been registered to the system. The Visitors may view and access information about the community in general, but they are not able to navigate through the system's functionality.

In order to achieve this, they have to register to the system, by completing a registration form, which will be approved by the Community Manager. The user main role is to request a service.

Once all the actors of the system and their interactions are identified, the next step is to model them and to associate them with their corresponding use cases, An M – UML Use Case diagram will help illustrating requirements and actors.

THE SYSTEM ARCHITECTURE

In this part we describe functionalities and architectural issues in the design of a Web Services communities management system. Our system can be seen as a mediator used to process users' queries and identify relevant share knowledge. When the query is received, the mediator analyzes the query and locates relevant sources, and presents the answer that is merged, assembled or redefined. Furthermore, it may keep, fully or in part, the result of frequent queries

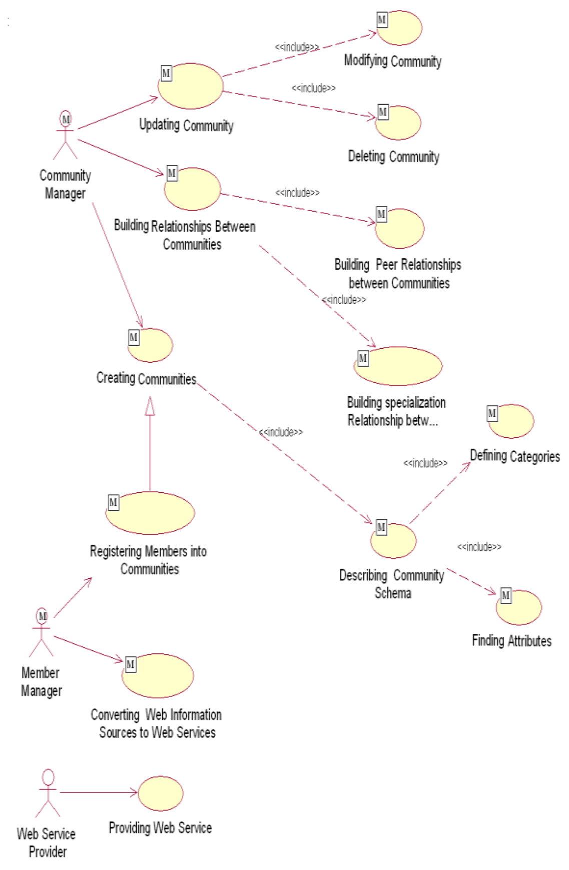

Fig .1 M-UML Use Case diagram for communities management

The mediator is in charge of community creation, community update, community members' participation and the research of the best provider for a requested service. The mediator architecture includes five main components. The Community Knowledge Base (CKB) where information about communities are stored, The User Queries Knowledge Base (UQKB) where user queries are stored and a Query Solver (QS) which main role is to extract and, route sub queries destined to specific communities. The Semantic cache (SC) saves the queries results in a queries cache for an immediate and future use of these queries. And the cache maintainer (CM) is responsible for the coherence and the availability of the cache.

The community Knowledge base (CKB)

The community Knowledge base is used in order to provide context for queries and is influenced by the current interaction of the user with the service. It represents all the entities that exist on the system, these entities are objects that become source of knowledge for the users of the Mediator .The Community Knowledge Base support the community creation, the community management and building relationships between communities.

The User Queries Knowledge Base (UQKB)

The User Queries Knowledge Base offers a space to a user in which he is able to retrieve information stored in the system, communicate or interact. The User Queries Knowledge Base performs the task of the description and the formalization of the query.

The Query Solver (QS)

The main functionality assumed by the query solver consists on processing the query ad answering it which requires locating Web Services capable of giving an answer to the query by routing the query among communities then querying individual community .The result of this step is finding the combination of members satisfying the query constraints

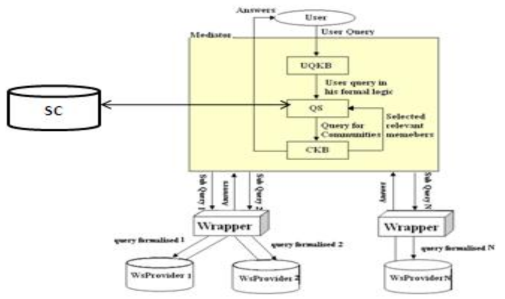

Fig.2 The system architecture

BUILDING COMMUNITIES

A community describes its ontology in terms of categories and descriptive attributes. For example, the health care community may have a category patient, which is described using attributes such as name, birth date, height, weight, sex etc.

The community description language

To provide formal semantics in describing the ontology, necessary for precise characterization of queries over the catalogs, we use a (concept) class description language that belongs to the family of description logics (Baader et al., 2003). The community ontology (also called community schema) is described in terms of classes (unary predicates) and attributes (binary predicates). Class descriptions are denoted by expressions formed by the following constructors:

Class conjunction (π), e.g., the description diagnosis π pathologies denotes the class of diagnosis which are instances of the classes' pathologies diagnosis

The universal attribute quantification (∀), for example, the description∀ Patient Diagnosis. Diagnosis the class of patients for which all the values of the attribute Patient Diagnosis are instances of the class Diagnosis (i.e., the data type of the attribute Patient Diagnosis is Diagnosis),

The existential attribute quantification (∃R), e.g., the description ∃ pathology denotes the class of patients having at least one value for the attribute pathology

A community is a container of Web Services of the same domain (e.g. Health care Community). It provides a description of desired services without referring to actual providers (e.g. Hospital). The schema of a community is described in terms of categories and descriptive attributes. For example, the community UrgentPatient may have a category Patients, which is described using attributes such as name, weight, height, etc. In fact, a community schema can be viewed as ontology (integrated schema) of its underlying services.

Community creation

To provide formal semantics in describing categories and attributes, we use the class description language. The users do not describe schemas and queries directly using description logic. Instead, a graphical interface is used to automatically generate these descriptions.

To create a new community, a community administrator creates the community ontology. S/he first defines the name of the community and the root category. The sub categories are then added and attributes are defined for each category.

Note that a sub category inherits all attributes of its parent category. As part of the community ontology, each category (respectively for each attribute) is annotated with a list of synonyms. After the initial ontology is defined, a description logic converter automatically generates the class descriptions of the given ontology. For example, the category Diagnosis in figure may be described as follows:

Diagnosis =Examination π diagnosisDate π ∃ Date π ∀ DiagnosisType.VARCHAR π ∃ DiagnosisType.

It states that the category Diagnosis inherits all the attributes of the category Examination, and has two additional attributes, namely Diagnosis Date and Diagnosis Type.

Being a web service, a community in the management system implements a standard set of operations which can be invoked by the user or the peers (e.g., add Category(), addPeer(),query Community() etc.

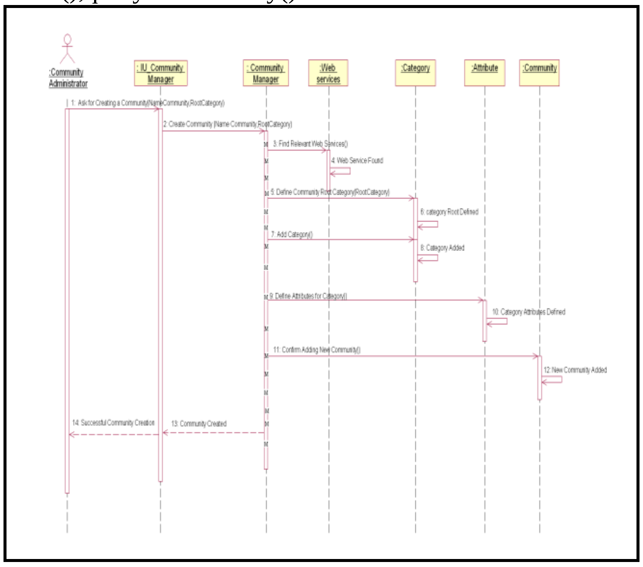
:Community Administrator
: IU_Community Manager
: Community Manager
:Web services
:Category
:Attribute
:Community
1: Ask for Creating a Community(NameCommunity,RootCategory)
2: Create Community (Name Community,RootCategory)
3: Find Relevant Web Services()
4: Web Service Found
5: Define Community Root Category(RootCategory)
6: category Root Defined
7: Add Category()
8: Category Added
9: Define Attributes for Category()
10: Category Attributes Defined
11: Confirm Adding New Community()
12: New Community Added
14: Successful Community Creation
13: Community Created

Fig.3 M-UML sequence diagram of community creation

The Members Subscription

Potential communities' members are of two kinds:

1. Web Services Provider who already has his catalog accessible via a web service.
2. Service Provider who needs to create a web service which accesses his service. For the latter, the mediator provides a Service Provider with functionality which is similar to the one used in creating communities. Using the provided functionality, the service provider can describe the service's ontology (in terms of categories and attributes).

When registering, the member first indicates which categories, in the community; the member's service belongs to. For each category selected, the member specifies what kinds of attributes are supported for the category (called member definition). The member definition from a member are also converted to the class descriptions and added to the community's ontology.

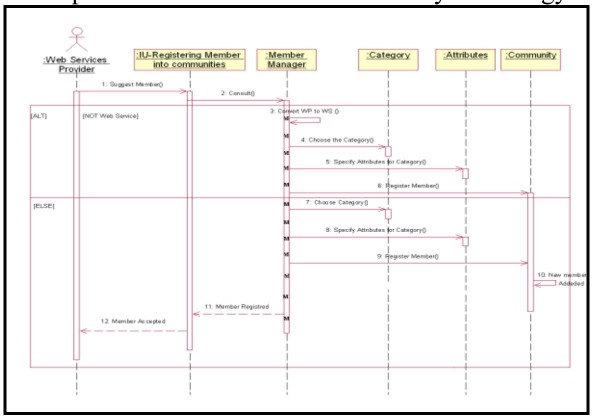

Fig.4 M-UML sequence diagram of registration into community

Communities Update

The changes that alter communities can take two main forms.

Community deletion The Community Manager deletes the community does not contain any Web Service. For this purpose he has to identify a community that users constantly leave without performing any further action .The Deletion is logic to allow users being members of other communities.

Community modification consists on adding or updating members or adding new Web Services manually or automatically

RELATIONSHIPS BETWEEN COMMUNITIES

Building Relationships

Relationships between communities fall into two types: Sub Community Of relationships and peer relationships.

Sub Community Of relationships represent specialization between two communities' domains (for example, Hospital is a sub community of Medical Institutions).

Peer Community Of When a user cannot find (or is not satisfied with) information from a community, she or he can refer to other communities that the community considers as its peers (for example, Patient is a peer community of Examination). We do not assume that the opposite systematically holds (for example, that Patient is a peer community of Examination). A weight (a real value between 0 and 1) attached to each PeerCommunityOf relationship represents the degree of relevancy as a peer. Communities can also forward queries to each other via a PeerCommunityOf relationship.

Relationship Update

Merging communities: Consist on identifying two subcommunities of the same supercommunity that are always accessed together (not viaPeerCommunityOf). This operation merges two communities' members, Ci and Cj, which have the same supercommunity. (An administrator uses this operation, for example, when she or he observes that users almost always access Ci and Cj together. So, merging these two communities will be beneficial because most users will not have to visit two separate communities each time.

Updating the weight of a PeerCommunityOf: This operation updates the relationship's relevancy according to two cases

Upgrading the weight of a PeerCommunityOf: Assume that many users navigate from community Ci, via a PeerCommunityOf relationship, to community Cj and submit a query to Cj. This indicates that the PeerCommunityOf relationship from Ci to Cj positively contributed to finding the target community, Hence the weight of the Peercommunity Of has to be upgraded.

Downgrading the weight of a PeerCommunityOf: Assume that many users who followed a PeerCommunityOf relationship and arrived at a community Cj and ultimately leave the community without performing any further action. This might indicate that cj is not relevant to these users.

THE QUERY PROCESSING BETWEEN COMMUNITIES

Users in our mediator will typically be engaged in two steps information research activities: navigating communities for services location and semantic exploration(e.g., get communities that are relevant to patient pathologies) then querying selected communities

Searching Relevant Communities

Users would have a specific task to achieve (e.g.,Diagnosis they wish to consult, a category of Examination they want to investigate) .We assume that they use the following strategy :

1. Start at the root (i.e.health care Community), or at a specific community (if they know the location of the community they want to query).
2. While (current community C is not the target community T) do
a) If any of the SubCommunity–Of relationships of C seems likely to lead to T,follow the relationship that appears most likely to lead to T.
b) Else, if any of the PeerCommunity–Of relationships of C seems likely to lead to T, follow the relationship that appears most likely to lead to T.
c) Else, either backtrack and follow SuperCommunity–Of relationship of C, or give up.

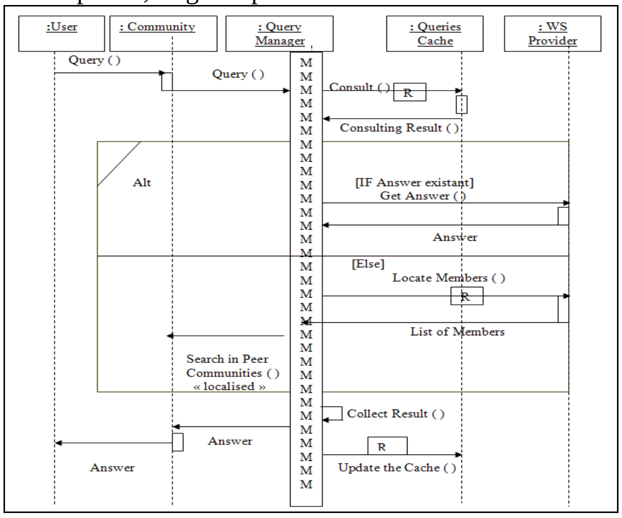

Fig.5 M-UML sequence diagram of query processing

Once the user has reached the target, s/he will submit a query to the target. If the user ends up in the same community again in step 2(a) or 2(b), s/he will follow a different relationship, since her/his reasoning of which relationship is likely to lead to the target has changed by then Querying selected communities

The Communities querying process follows the following steps:

The User uses a community to express queries; he submits the query to the current community. SubmitQuery (userId, QueryQId) . For example, suppose that a healthcare unit from a hospital A needs to get information about a patient from a patient community which has been encoded in a different hospital B, and that it has to check the updatability of that information according to the system used in the hospital a in order to provide the necessary (and minimal) information to the units that carry on tests (so that they can schedule them), to the administrative unit (so that it can manage the patient's case), to the patient herself (so that she receives a readable but precise résumé of her case), etc.

The Query Manager takes as input the UserId, the QueryQ and the community definition C (userId, QueryQId, C) .It sends directly the query to the Queries Cache before its processing. The QC identifies the part of the query answer that exists Qexistant inside it (user Id, Qexistant, Qanswer).Then return it to the QM. If the answer satisfies most of the constraints expressed by the query then the result is delivered to the user

Else the QM continues the processing of the rest of the query. It rewrites Qnonexistant the others parts that cannot be answered by the QC (userId, Qnonexistant, C). The community will collaborate with peers to identify any external members who can answer this part of the query.

Answers are collected by sub community then community and are added to the answers given by the QC then sent to QM which delivers the result to the user.

CONCLUSION AND FUTURE WORK

In this paper we proposed a design of a system able to manage Web Services Communities taking into account many tasks such as discovering and updating communities then building relationships between them etc. Moreover, these communities are built with the goal to be queried transparently and easily by users which aim to satisfy their informational and physical needs in a satisfactory time and in a pertinent retrieval .For an efficient query processing we developed an advanced caching mechanism to reduce the query response time.

To provide for availability of the cache even if a community disappears, a replacement mechanism of the affected community by one of its peers has to be envisaged. The idea of the cache maintenance seems to be very promising and will be detailed in our future work.

REFERENCES

Baina,K. Benatallah,B. Paik, H. and Toumani, F. (2004). 'WS-CatalogNet: An Infrastructure for Creating, Peering, and Querying e-Catalog Communities', in Proc. of VLDB , Toronto, Canada, demonstration paper.
Benatallah,B. Hacid, M.-S .Paik, H. Rey, C and Toumani, F (2004). 'Towards Semantic-driven, Flexible and Scalable Framework for Peering and Querying e-Catalog Communities', Information Systems Journal,Special issue on semantic web services.
Benatallah, B .Hye-young ,Pan d Toumani, F .(2004).'Towards Self-organizing Service Communities', IEEE transactions on systems, man and cybernetics.
Kassem, S and El Morr,C .(2003). 'M -UML an extension to UML for the modelling of mobile agent based software systems', information and software technology
K.Yamada, K. Nakakoj and K. Ueda.(2004). "A Mutli-agent Systems Approach to Analyze Online Community Activities" The Fourth International Conference on the Advanced Mechatronics
Koch,M and lacher, M.(2000). 'Integrating Community Services-A common Infrastructure Proposal'.
O'Murchu, I., Breslin, J.G., and Decker, S.(2004). "Online Social and Business Networking Communities". In Proceedings of ECAI Workshop on Application of Semantic Web Technologies to Web Communities.