Smart Places: Multi-Agent based Virtual Community Management System

Muhammad Fahad*, Olivier Boissier#
*Ecole Nationale Supérieure des Mines
FAYOL-EMSE, LSTI,
F-42023 Saint-Etienne
Muhammad.Fahad@univ-st-etienne.fr
Olivier.Boissier@emse.fr

Pierre Maret*, Christophe Gravier*
*Université de Saint-Etienne
10 rue Trél刻苦e
F-42023 Saint Etienne, France
Pierre.Maret@univ-st-etienne.fr,
Christophe.Gravier@telecom-st-etienne.fr

ABSTRACT
Virtual Communities (VC) are synergetic digital spaces to build collaborations and exchange knowledge between people sharing similar interests or goals. They respond to human needs such as information sharing, friendship, and recreation. In this paper, we present a VC Management System, based on a Multi-Agent Oriented Programming (MAOP) approach. Using the MAOP JaCaMo platform, we make possible to install a decentralized and open management of those communities in a context of smart cities. Participants of a virtual community are each supported by a Jason agent that encapsulates the logic and the control of the user participation to a VC (such as publishing posts, notifying members, etc.). A set of CArtAgO artifacts provides the basic functionalities and operations giving access to the basic functionalities for knowledge exchange in such VC. Agents can exploit those artifacts to execute their tasks while achieving their individual and collective goals. The global coordination strategies enforcing behaviors of the participants of a VC are expressed as multi-agent normative organizations managed by the Moïse framework.

Categories and Subject Descriptors
D.3.3 [Programming Languages]: Language Constructs and Features – abstract data types, control structures.
I.2.11 [Artificial Intelligence]: Distributed Artificial Intelligence – Intelligent agents, Multi-agent systems, Languages and structures

General Terms
Management, Design, Experimentation, Human Factors, Languages.

Keywords
Multi-Agent programming, Virtual communities, Information Exchange, Jason, Cartago, JaCaMo

1. INTRODUCTION
Virtual Community (VC) (also called “Online Community”) is the gathering of people, in an online space where they join, communicate, and get to know each other better over time. With the passage of time, VCs become as a synergetic way to exchange knowledge within group of people with similar interests. VCs were originally designed to be agnostic of spatio-temporal constraints. Instead, they are nowadays spreading under ad hoc spatio-temporal constraints in the physical environment. VCs expanded from the paradigm to offer online spaces for users to communicate to providing a digital medium for people in the same area, at the same time. It mostly occurred thanks to (or “because of”) the advent of pervasive computing in our lives. Mobile and ambient computing lead VCs incarnation to shift from digital environments to the real world. That is the reason why there is a strong need of decentralized and open ways for managing these VCs in such settings. For example, a community within the university can serve different types of information exchanged by the students, professors, administration, etc. Or a community server can be situated at some visiting place like ‘Eiffel Tower’, for managing the visiting people, history of the place, controlling exhibitions, reporting improvements, etc.

To tackle with such requirements, we turn to multi-agent technologies that showed success in various application domains, where different autonomous decision-making entities (agents) have to communicate, exchange knowledge and cooperate in order to achieve individual and/or collective objectives [1]. Using a Multi-Agent approach for supporting VC, agents act as a personal assistant on the behalf of each member of a community. The agent perceives knowledge from the communities of individual interests and acts upon the communities to meet their design goals. Thus, agents bring the appropriate people having common goals or interests together and to share their knowledge with each other at ease.

In this paper, we present a multi-agent based VC Management System, where agents can help their users in accessing, exchanging and managing information within the community of interest or group of people. To realize the goals of our VC Management System, we have used the JaCaMo® platform that combines three separate open source technologies, i.e., Jason, CArtAgO and Moïse for programming MAS. Each of these programming technologies has been used for building multi-agent systems.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

WAC’12, April 16, 2012, Lyon, France.
Copyright © 2012 ACM 978-1-4503-1189-2/12/04…$10.00.

1 http://www.jacamo.sourceforge.net
applications for a number of years. Jason has been used for programming semantics of a variant of AgentSpeak and provides many user-customizable features for the development of Multi-Agent Systems [2]. CArtAgO (Common ARTifact infrastructure for AGents Open environments) has been used for programming environment artifacts. By using this infrastructure, agents dynamically create and use artifacts as a fundamental building block for achieving their activities. CArtAgO Artifact is the functional-oriented and stateful entity comprise of controllable and observable properties, created by the agent and used as a basic resource by the multi-agents [3]. Lastly, Moise provides infrastructure for programming multi-agent organizations [4].

With the combination of these technologies, JaCaMo covers all levels of abstractions that are required for the development of sophisticated multi-agent system, such as, Virtual Community Management System. In addition, the approach offers many advantages, such as decentralization of the community management, personalized automatic management and discovery of communities, and flexibility so that any agent can create its own community. We have investigated this approach with the idea to provide Virtual Community Management System for smart cities.

The rest of the paper is organized as follows. Section 2 provides a discussion on a scenario description of a smart city. Section 3 presents our Virtual Community Management System. Section 4 elaborates different functionalities of our system for smart places use case. Section 5 discusses related work. Section 6 concludes the paper and shows our future directions.

2. A Smart City – Scenario Description

In this section, we explain a use case about a virtual community management platform for a smart city.

The smart city is equipped with community servers situated at different places or buildings of the city. The purpose of community servers is to manage different communities and to provide services to inhabitants. Communities are created on the fly by inhabitants of the city or by any community service. A community server is associated to a physical area defined by geographical coordinates (for instance a circle around the server). These servers can be situated at various places of the city. For instance, in a Fashion Show for the management of the community of the people who participated and watched the fashion show, interested in sales and purchase of the displayed fabrics, collection of votes and choosing the best models from the participants. It can be situated in a theater for the management of the community of people interested in watching the theater, interested in the history of the story behind the specific theater and certain drama or topic behind the theater, agreeing to let them informed about the next shows matching their interests, collection of their opinions about the present and upcoming theater shows. It can also be situated at any place of the city for the management of the community of citizen pushing information related to the improvement of the friendship in that place, management of the community of the citizen pushing information to the city services to signal them things to improve and repair, etc.

The community servers manage communities by offering different functionalities. These functionalities are available only to users situated in a given neighborhood of the server (i.e. a certain distance, community area). A user situated in the right area can interact with the community server via his/her community assistant (or personal agent) that is installed on his/her smart device and exploit services such as create a community, join a community, publish information in the community, get information from the community, etc. These actions and community services are not feasible when the user leaves the area. Since the user may not stay in the VC for a long time, or since he/she may not be ready to interact with his/her device at that time, users delegate to the smart device the management of these different actions according to the topics that he/she is interested in. He/she expresses in advance the information he/she is ready to share a priori with other users.

3. Virtual Community Management System

The Virtual Community Management System (VCMS) that we have developed, implements a generic approach for creating “mass” of places of local exchanges and local knowledge bases in the context of smart places. It provides different services to build and manage VCs, as explained in the following subsections.

3.1 Architecture of VCMS

VCMS is based on a multi-agent architecture that enables community assistants (participants) to meet, share and gain quick and efficient access to the information of their interest. In VCMS, virtual communities are realized by means of a set of Jason agents encapsulating the user profile and the logic and control of the specific operations involved in the community pack: community management (e.g. joining/leaving, creating/deleting a community), information sharing (e.g. publishing posts, notifying members, etc.). The layered architecture of VCMS is shown in Figure 1.

![Figure 1. VCMS Architecture](image-url)
The agents accomplish these actions by using a set of functionalities that are made accessible by the communities they belong to. Agents can exploit them to achieve their individual and collective goals. These tools are implemented by means of artifacts developed in CArTAgO. Agents lookup the artifacts in the different workspaces affected to each community and exploit the operations offered by each of the artifacts to achieve their desired tasks. The structure and functioning of each community is declaratively specified in an organization specification where owner, member or participant roles are defined. A community consists of a domain of interest, an owner, a message buffer, and an initially unspecified number of member agents, which have certain interests over the community. By deciding to play one of the roles, according to the norms stated in the organization, agents may create a community (owner role) or participate to it (member role).

3.2 Community Tools for VCMS

The Virtual Community Management System (VCMS) allows the participants to build different kinds of communities by dynamically instrumenting their communities with different tools to manage the information shared between participants according to their requirements and needs. All the information exchanged by agents participating to a community is stored in these tools. Currently, the VCMS provides four basic classes of community tools as follows.

3.2.1 MailBox

This type of VC tool allows any member of the community to send and receive messages to/from other members of the community. Once an agent gets its membership to a VC, it will receive all the messages and updates posted on the community automatically. For instance, when a late comer joins the VC, all subscriber agents receive a message about its subscription to the VC, so that they know and can initiate further communication. An illustration of this class of VC is the “Catwalk” on a Fashion Show, in which an organizer wants to communicate with models participating in a catwalk show, and models themselves want to communicate with each other. An ad-hoc community is therefore shaped under a publish/subscribe paradigm for information exchanges.

3.2.2 Participatory Forum

This type of VC tool allows registered members to participate in the community by different services, such as reading and writing posts on the VC. By getting registrations for the community, members will receive all the posts and updates automatically. There are many options that can be manipulated with this type of VC, i.e., Classic, Duration, Moderation, Bound, etc. For example, the Bounded BlackBoard Community tool allows the creator of the VC to set the limit of the blackboard, which decides the content of the board. It lists messages on the board in an ante chronologically order. When reaching the specified limit, the buffer of the message board acts like a First-In First-Out (FIFO) manner and deletes the last message of board to place the new message on the top of buffer. Example of such community can be “Contest Board” on a Fashion Show, where management displays names of models presenting the fabrics and status of designers with their votes or likings by the viewers participating in the show. The board has limited capacity and only newly important messages are placed and old messages are deleted for promoting information lookup at ease.

3.2.3 Information Dispatcher

In this type of VC tool, only the VC owner can disseminate information. Members of the VC automatically receive messages posted on that tool. Example of such a community can be “Catwalk Schedule” on a Fashion Show, where only organizers can post messages about the schedule, names of selected participants, updates about displayed fabrics and brands, etc., for the registered or regular members. These members can only read the post, but they cannot post any information over the VC.

3.2.4 Personal-Box

In this type of VC tool, only the VC registered members can disseminate and share information over the community. Other community assistants that are non-members of the community can consult this type of tool, but they are not allowed to post messages without having membership. The owner of the community has the only right to give membership to other community assistants to get involved in the community. Example of such community can be “Model of the Month” on a Fashion Show, where only registered users can cast a vote or judges can post their opinions for choosing the best model from the participants.

4. VCMS for Smart Places Use Case

The VCMS allows participants with different functionalities for building and manipulating virtual communities as. A participant can create his/her own community or get the membership of existing communities. Using the options “Join Community” and “Leave Community”, one can join and leave the community if one fulfills the criteria of joining or leaving. Using the option “Show Communities”, one can see all the present communities with the list of topics associated with each of the community.

The VCMS allows the management of the community content within the topics or the subject of the community. Such management of community content allows flexibility in posting, searching and disseminating information within the community. With options such as ‘Add Topic’ and ‘Show Topic’, participants can add a new topic in the existing community and see the existing topics under one community. It allows participants to post messages (i.e., raw content in the form of text, picture, etc.) over the community. The members are free to choose the topics of interest for the community and allow one to post the messages related to the topics over the community for the other participants. With options such as ‘Add Message’ and “Show MessageBoard”, participants can add a new message in the existing community and see the existing messages over the specific topics.

The VCMS allows participants to follow different communities of his/her interest. When any member joins the community, it will receive all the posts automatically in his mailbox depending on the community type and its rules. When calling the functionality “My mailbox” community assistant updates its owner with the list of messages about the communities in which he is registered or some other persons have mailed him. Besides this, VCMS allows participant to manipulate different informations of his interest. Once a user enters his topics of interests, his personal community assistant processes match making mechanism and explores the information over communities. Then, it brings its user with some recommendations beneficial for him. It is not necessary for the user to be a member of that community, only information about his/her interests in his/her profile is sufficient to receive recommendations. With option “My Recommendations”,

community assistant updates its owner with the list of recommendations about the things of his/her interest.

It is possible for participants to search for existing communities with different criteria. One can search the community by name or topic. The VCMS extracts and displays the information about the community if it exists by that name or topic. Another option for community lookup is the semantic search for the community with the help of WordNet. In this case, it generates the SynSet of the keyword entered by the user as input and matches with the existing names of communities and/or existing topics within the communities, and user is prompt with the matched results if any. We consider semantic search as a vital mechanism for our virtual community management, hence several extensions are going to be made in the near future. These features, such as autonomy on the mobile application and semantic reasoning capabilities for the end-user decisions, differentiate it from the existing mobile applications.

5. Related Work

P. Maret and J. Calmet developed a bottom-up agent-based approach for the knowledge management using virtual communities [5]. They developed a prototype named as Virtual Knowledge Communities based on the Jade system, which is a Java based software development framework that conforms to FIPA standards for intelligent agents. Z. Zhao et al. developed an agent based SymposiumPlanner system to promote topic-oriented collaboration between the distributed members of a virtual community [6]. It exploits Rule Responder that is a multi-agent system for collaborative team and community support on the Web. A semi-autonomous rule-based personal agent assists each of the community members. Their agent infrastructure is based on the Semantic Web rules that help to capture and make cooperate for different aspects of the member’s derivation and reaction logic. SymposiumPlanner supports the RuleML Symposia by coordinating personal agents that assist the symposium chairs, intelligently answering questions from people interested in the symposium.

A. Sorici et al. developed an adaptive room governance application in the context of smart co-working [7]. They focused on MAS organizations based on Moise framework (within JaCaMo platform) to establish a precise and efficient level of management for the room allocation. C. Toledo et al. used JaCaMo platform to develop a concrete agent-based architecture to proactively supply knowledge to knowledge-intensive work flows by integrating the Business Process Management and Knowledge Management infrastructures [8].

6. Conclusion and Future Work

In this paper, we present Virtual Community Management System that is built by using the JaCaMo platform. It implements a customizable approach to the creation of different communities, with different possible functions and modes of exchanges. It enables VC to be a place where agents can meet and share knowledge with other agents who share a similar domain of interest. We find JaCaMo as a power technology for building complex multi agent system encapsulating three different technologies. Especially, the agent environment programming as CArtAgO artifacts provides the set of basic API for agent platforms to work within artifact-based environments. It builds VCs as artifacts, that are treated as first-class entities representing resources and tools that agents can dynamically instantiate, share and use for their desired objectives. In addition, due to its build-in capacity, it allows to interconnect different VCs. The development and execution of artifact-based environment structured in open workspaces (possibly distributed across the network) where agents of different platforms belonging to several virtual communities can join and work together. Our ongoing research is to handle the organization of different communities and authorizations to agents with Moise framework using JaCaMo platform. Another is to test and deploy our VCMS on android based smart device using JaCa-Android platform.

In future work, we plan to provide an ontology of topics to the system. While it is too challenging to provide an ontology of topics we do not know yet, this ontology could be a shared specialization of topics co-constructed by all the users who are adding topics to the VCs. This can be simply achieved by asking the user, when they are adding a topic, in which part of the topic tree they want to insert their new topic, if it already not exists in the topic tree. To avoid free texting, system will restrict the user to choose a WordNet sense to have a clear interpretation of a topic. Would every user contribute to this ontology, we could make recommendations of VCs and topics using expansion of topics based on these subsumptions relationships, as they explicit narrower/broader relationships between topics created by the users.

7. ACKNOWLEDGMENTS

This work has been supported by the Conseil Général de la Loire, France.

8. REFERENCES