International Conference
FROM DELC TO VELSPACE
Plovdiv, 26–28 March 2014

AJTEMPURA – FIRST REALIZATION OF C3A MODEL

Vladimir Valkanov

Abstract. *The paper provides a general description of a model for context-aware agent architecture (C3A). The approach adopts the definition of context and context-awareness given by Dey. The C3A model aims at creating smart virtual spaces. The applicability of the model is demonstrated by the development of an agent-oriented application.*

Key words: Context-aware architecture, Formal models, Smart space, Intelligent agents, Agent-Oriented architectures, Tempura.
2010 Mathematics Subject Classification: 68T42 Agent technology

INTRODUCTION

One of the main characteristics of the modern systems today (i.e. eLearning) is the 'anytime-anywhere-anyhow' delivery of electronic content, personalized and customized for each individual user. To satisfy these requirements new types of context-aware and adaptive software architectures are needed, which are enabled to sense aspects of the environment and use the information to adapt their behaviour in response to changing situation. From a network of computer networks the Internet has transformed into the Internet of Things [1], where people and everyday objects can be assigned identifiers. These identifiers can be created and managed by computers. Within the infrastructure of the current Web, the new Symantec Web [18] is rising making all sorts of informational resources addressable by the existing identification protocol URI. A model for development of context-aware software architectures, known as Context-Aware Agent Architecture (C3A), is presented in the publication. There are a lot of definitions for context-awareness, the first attempts to give satisfactory definition are related to application of mobile devices and reading users' position. In the specialized literature it is presented as the term context-aware computing is used to describe a software capable of rendering an account of users' location [10] or as the ability of a program or device to sense various states of its environment and itself [4]. We adopt Dey's context definition for developing the C3A model. Dey [3] criticizes the above definitions in two ways: context is defined by examples, i.e. by enumeration of various cases, and context is defined by synonyms, mainly as environments or situations. In his opinion, these definitions should concern development of context-aware applications. In this sense he proposes

a more common definition whereby context is any information that can be used to characterize the situation of an entry. An entry can be assumed as a person, a place, or an object that is considered relevant to the interaction between a user and an application, including themselves [10]. In this point of view if a system which provides relevant information and/or services to the user, where relevancy depends on the user's task, it is called context-aware system. An implementation of the C3A in real software demonstrating the features of the model is also presented in this paper.

C3A MODEL

The model aims to propose a framework which can be used for implementation of context-aware applications for various domains. A context-aware architecture includes autonomous intelligent components which can: operate in a changing environment; detect, identify and localize changes (events) in the environment; initiate corresponding compensatory actions depending on the types of changes (events). A compensatory action is defined as an activity (functionality) triggered as a reaction to the occurrence of some event (change in the environment). Two basic compensatory actions are: *Adaptation* – before the information resources (electronic services, electronic content) are provided, they can be modified so that they reflect as completely as possible the specifics of the change or event; *Personalization* – before the information resources (electronic services, electronic content) are provided, they can be modified so that they reflect as completely as possible the users' desires, intentions, background, etc.

Using C3A, a formal presentation is proposed which reflects the spatial and temporal aspects of: virtual space structure and building components; relations between the components; operations allowed in the space; control of the processes taking place in the space.

In the model, the fundamental notion is this of ***smart space.*** In the smart space, ***services*** will be provided through autonomous intelligent components, known as *agents*. The agents are with 'limited rationality', which in this case means:

- They operate with limited capacity (resources) in order to plan, predict, make choices and execute actions;
- They have partial (local) control and impact on the smart space.

Due to the limited rationality, we would assume that:

- The set of services, provided in the space, alters dynamically;
- There is a *minimal functionality* (minimal set of services), that the space cannot operate without.

The set of all agents are potentially operating in the smart space. Due to the limited rationality we assume that each agent operates in its own subspace (known as *agent's range*) operating as the agent's environment. Agents can operate and

impact only within their own range. In this sense, the overall smart space is built as a union of agents' ranges.

Various types of agents and services are distinguished in the smart space. The set of agents is decomposed as two groups – persistent and operational agents. Respectively, the set of services can be decomposed as a union of three mutually disjunctive subsets. This decomposition characterizes the following types of services:

- services that manage the minimal functionality of SmartSpace;
- services performing the compensatory actions;
- services, which include three special operations which are used to create and remove operational agents.

Here, the two types of agents are presented shortly. Persistent agents (*PA)* includes agents, known as *persistent agents,* which are always available in the space. These agents are used to:

- Provide the minimal functionality of the space;
- Identify and localize the events which take place in the space;
- Generate operative agents;
- Remove operative agents.

The persistent agents' behavior is defined with genetic function, which allows them to generate operational agents (*OA*) dynamically when it is needed. An operative agent provides some kind of compensatory actions as a reaction to the events identified in the range of the generating persistent agent. After completing the compensatory actions, the operative agent will be removed (or self-removed).

The events which take place in the space are introduced with two basic characteristics. *Location* and *time of occurrence* are the two basic features of the events. C3A examines the locations of the events' occurrence as unique subspaces so that $SmartSpace^{E}$ presents the overall SmartSpace in terms of the locations of expected events' occurrences. The events are always viewed as *atomic primitives,* and they can be combined to make more complex structures such as situations, scenarios, intervals. An interval is the sequence of events ordered in time where an event can be identified at any time – past, present and future.

As already pointed out, the persistent agents perform two different functions. First, they provide the minimal functionality in the SmartSpace. Second, they can identify the occurrence of certain events in its ranges and dynamically generate appropriate operative agents, respectively, which in turn execute the necessary compensatory actions.

The general lifecycle of a context-aware agent-based software architecture compatible with C3A is shown (in pseudocode) in Fig. 1. Presented in time, the operation of the architecture looks like a pulsating core (implementing the minimal functionality of the SmartSpace) which periodically 'inflates' in different directions (depending on the changes in the environment) and again 'deflates' to its usual size (minimal functionality).

AJTEMPURA AS A C3A APPLICATION

Application of the C3A model for development of real software will be demonstrated in this section. E-Learning environments providing electronic educational services are becoming an integral part of modern education. In the Faculty of Mathematics and Informatics at the University of Plovdiv, an infrastructure is being developed, known as Distributed e-Learning Center (DeLC), as a response to the need for supporting learning using modern information and communication technologies[14,11,13]. The center aims to provide adaptive and personalized e-learning services and teaching content located on physically separated servers. DeLC is a dynamic network structure consisting of [12]:

- Nodes – these operate as repositories of services and content;
- Relations – these specify various kinds of dependencies arising during an education experience.

The nodes can operate independently or dynamically link to each other, making complex virtual structures, called educational clusters. In the current DeLC versions, two educational clusters are built. The first cluster called MyDeLC is used to organize and perform e-learning by allowing fixed access to the services and electronic content via a specialized educational portal [15,16]. The second cluster, called InfoStation cluster offers a three-layer architecture which allows mobile access to services and informational resources via intelligent wireless access points (called Information Stations – ISs), situated around the university building [17,7,6]. In 2013 we have started a new project that aims at the transformation of DeLC in a new infrastructure called Virtual eLearning Space (VeLSpace). In this space, the context-aware provision education services and teaching content will be supported by autonomous intelligent agents with 'limited rationality', i.e. changes can be identified only locally within the ranges of the agents positioned there. Each event causes a change in the local state of the range. In this way the state of the overall VeLSpace (global state) depends on the local states ordered in time. In order to manage the global state, a suitable formalism known as Interval Temporal Logic (ITL), was chosen as a theoretical model. Interval Temporal Logic [9] is a kind of temporal logic for describing time-dependent processes. Tempura interpreter [8] is an executable subset of ITL which uses the ITL syntax to identify time as a finite consequence of states. The existing version of Tempura was created in the C language [5] and repeatedly amended and expanded with new functionalities. Since the VeLSpace is built by separate intelligent autonomous components where the electronic services are equipped with their own operational agents, we created a new agent-oriented version of Tempura, called AjTempura, which can be easily integrated in VeLSpace while maintaining the space's homogeneity.

The transformation of the original Tempura interpreter was an iterative hand-made process which consists of three basic steps. The software result of each step was tested and its run-time was compared with the original Tempura interpreter. Firstly we made a direct translation from C to Java code without changing the

imperative structure of the interpreter. The second step was a refactoring process which aims at creating an object-oriented version of Tempura called jTempura [19]. The final step was creating an agent-oriented version AjTempura. The AjTempura architecture is an implementation of the C3A model which is briefly described below.

The minimal functionality of AjTempura is implemented through two persistent agents – IOAgent and TempuraAgent. Both agents listen to the occurrence of events in VeLSpace. According to the kind of the identified event, an operative agent will be generated. This new agent has to analyze the event in more details and to initiate a corresponding compensatory action. AjTempura is implemented in development environment JADE [2]. The operating of the architecture is presented in the following diagrams.

Besides the two persistent agents, another one, known as Sniffer, is of key importance for tracking communication among agents. Its role is to provide a simple visual means for presenting the consecutive exchange of messages between the agents in a system. In its nature, the Sniffer agent draws diagrams similar to the Sequence diagrams in the UML language. The starting condition of this diagram, at the AjTempura start-up, can be seen in Fig. 2. In the left part of the figure, there is a list of the existing agents, among which there is a representative of the persistent IOAgent and a service DF agent. In accordance with AjTempura's lifecycle, the IOAgent starts sending periodic queries to DF for the presence of agents from the TempuraAgent type. The messages are sent every second until the reception of a response which contains a list of the present agents from the wanted type.

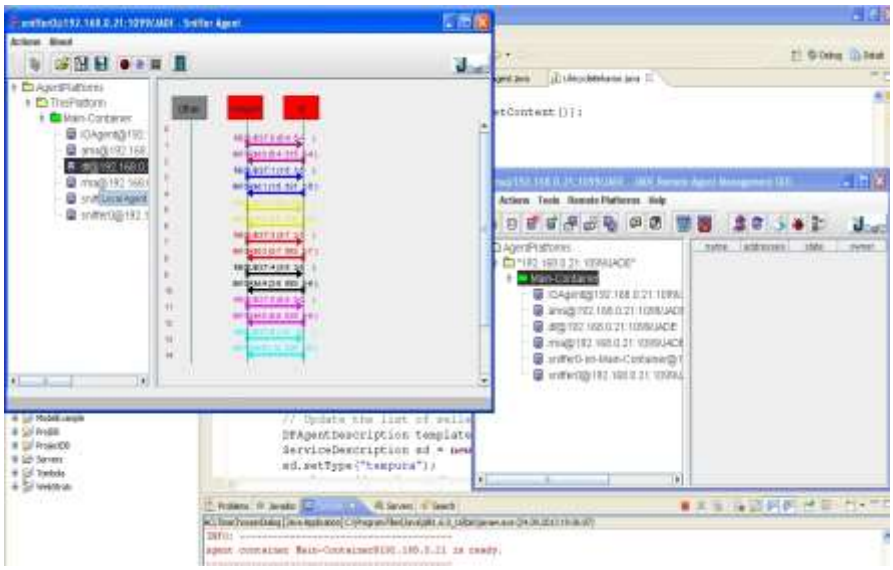

Figure 1. Starting situation of the Sniffer agent

At the appearance of an agent from the TempuraAgent type in the DF list, an exchange of messages between this agent and IOAgent immediately takes place, which leads to the generation of an operative interpreting agent. The appearance of

the necessary interpreting agent marks the start of the exchange of ITL sequences between this agent and IOAgent until the IOAgent goals are met. The communication between the entire set of agents can be seen in the following state of the diagram, generated by the Sniffer agent.

As a result from the appearance of the TempuraAgent and its communication with IOAgent, a new operative agent IA_1 has been generated. Its name is unique, and each following similar agent will be generated with a consecutive number. The operative agents exist until the IOAgent sends a 'done' message, i.e. confirmation for completion of the compensation action. In this case, this means that there are no more ITL sequences for processing and the IA_1 is redundant. After that the operative agents can be removed or self-removed .

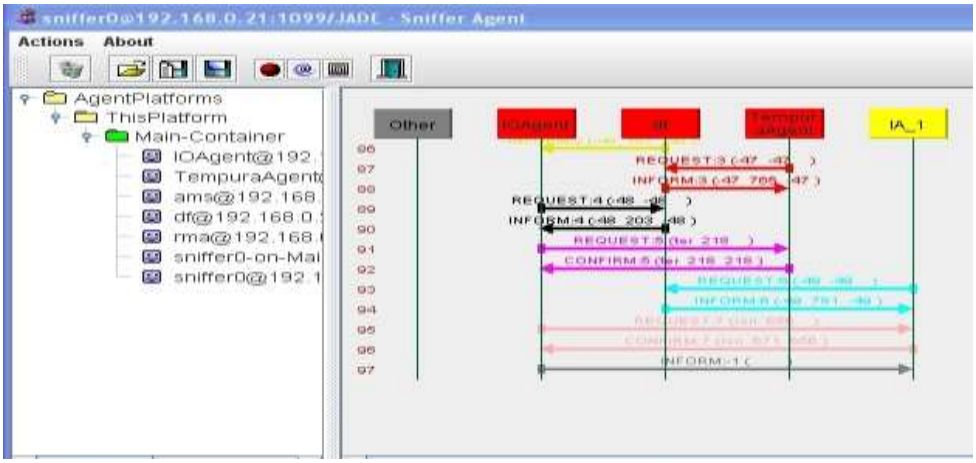

Figure 2. Third view from the Sniffer agent

CONCLUSION

The current state of the C3A model is presented in this paper. The research continues to expand and refine the existing version of the model. There are various issues being tackled. To name a few, for example, conflict resolving in overlapping or identical subspaces, saving traces of the removed operative agents as subintervals (the past), formalizing the interfaces between agents and education services, formalizing the notion of adaptation and personalization. The model extensions will be prototyped and examined in the VeLSpace.

ACKNOWLEDGMENT

The author wish to acknowledge the support of the Science Fund of the University of Plovdiv "Paisii Hilendarski" (Research Project Ref. No. NI13-FMI-02).

REFERENCES

[1] Ashton, K., 'That Internet of Things' Thing, in the real world, things matter more than ideas, *RFID Journal*, June 22, 2009.

[2] Bellifemine, F., G. Caire and D. Greenwood, *Developing Multi-Agent Systems with JADE*, Wiley, 2007.

[3] Dey, A., Understanding and Using Context, *Personal and Ubiquitous Journal*, Vol. 5, No. 1, 4–7, 2001.

[4] Dey, A. and G. Abowd, Towards a better understanding of context and context-awareness, *Proceedings of the 1st international symposium on Handheld and Ubiquitous Computing*, New York, 2000.

[5] Hale, R., *Programming in Temporal Logic*, PhD Thesis. Crambridge, England, Cambridge University, 1988.

[6] Ganchev, I., S. Stoyanov, M. O'Droma and I. Popchev, Enhancement of International IEEE Conference on Intelligent Systems, *2nd International IEEE Conference on Intelligent Systems*, Varna, 2004, 359–364, ISBN:0-7803-8278-1.

[7] Ganchev, I., S. Stoyanov, M. O'Droma and I. Popchev, An InfoStation-Based University Campus System Supporting Intelligent Mobile Services, *Journal of Computers*, Vol. 2, No. 3, May 2007, 21–33.

[8] Moszkowski, B., *Executing Temporal Logic Programs*. Cambridge, England: De Montford University, 1985.

[9] Moszkowski, B. and Z. Manna, Reasoning in interval temporal logic, *Proceedings of the ACM/NSF/ONR Workshop on Logic of Programs*, 1984, 371–383.

[10] Pascoe, J., Adding generic contextual capabilities to wearable computers, *2nd International Symposium on Wearable Computers*, 1998, 92–99.

[11] Stoyanov, S., I. Ganchev, I. Popchev and M. O'Droma, An Approach for the Development of InfoStation-Based eLearning Architecture, *Compt. Rend. Acad. Bulg. Sci.*, Vol. 62, No. 9, 2008, 1189–1198.

[12] Stoyanov, S., I. Ganchev, I. Popchev, M. O'Droma and R. Venkov, DeLC – Distributed eLearning Center, *1st Balkan Conference in Informatics*, Thessaloniki, Greece, ISBN: 960-287-045-1 , 2003, 327–336.

[13] Stoyanov, S., I. Popchev, E. Doychev, D. Mitev, V. Valkanov, A. Stoyanova-Doycheva, V. Valkanov and I. Mitev, Educational portal, *Cybernetics and Information Technologies (CIT)*, Vol. 10, No. 3, 2010, 49–69.

[14] Stoyanov, S., I. Popchev, S. Ganchev and M. O'Droma, From CBT to e-Learning, *Information Technologies and Control*, Vol. 4, 2005, 2–10.

[15] Stoyanov, S., A. Stoyanova-Doycheva, I. Popchev and M. Sandalski, ReLE – A refactoring Supporting Tool, *Compt. Rend. Acad. Bulg. Sci.*, Vol. 64, No. 7, 2011, 1017–1026.

[16] Stoyanov, S., I. Ganchev, I. Popchev and I. Dimitrov, Request Globalization in an InfoStation Network, *Compt. Rend. Bulg. Acad. Sci.*, Vol. 63, No. 6, 2010, 901–908.

[17] Stoyanov, S., I. Ganchev, M. O'Droma, H. Zedan, D. Meere and V. Valkanova, Semantic Multi-Agent mLearning System, *Semantic Agent Systems: Foundations and Applications*, Book Series: Studies in Computational Intelligence, Vol. 344, M. T. Kone, M. A. Orgun A. Elci, Ed.: Springer Verlag, 2011, ISBN: 978-3-642-18307-2.
[18] Berners Lee, T., J. Handler and O. Lassila, The Semantic Web, *Scientific American*, Vol. 284, May 2001, 34–43.
[19] Вълканов, В., *Контекстно-ориентирано управление на електронни услуги*. София, България: Академично издателство „Проф. Марин Дринов", 2013, ISBN 978-954-322-701-3.

Пловдивски университет „Паисий Хилендарски"
Факултет по математика и информатика
4003 Пловдив, бул. „България" № 236
Владимир Николаев Вълканов
vvalkanov@uni-plovdiv.net

AJTEMPURA – ПЪРВА РЕАЛИЗАЦИЯ НА C3A МОДЕЛА

Владимир Вълканов

Резюме. *Статията представя описание на модел за контекстно зависими агентни архитектури (C3A). Подходът приема дефиницията на контекст и контекстна зависимост на Дей. Приложимостта на модела е демонстрирана чрез разработването на агентно-ориентирано приложение въз основа на C3A модела.*