AMBIENT-ORIENTED MODELING IN VIRTUAL-PHYSICAL SPACE WITH INTEGRATED DOMAINS

Todorka Glushkova, Konstantin Rusev, Stanimir Stoyanov

Abstract. *The article discusses an approach to modeling in Virtual Physical Space with complex services based on integrated domains. The approach is particularly suitable for supporting the process of developing CPSS applications that provide comprehensive services to their users. The application of the approach for developing a specific model through a specially developed CCA editor is demonstrated.*

Key words: ambient-oriented modeling, Calculus of Context-aware Ambients (CCA), Virtual-Physical Space (ViPS).

1. Introduction

Modern realities in the digital age place new horizons in the creation of intelligent spaces in which people, digital devices, objects from the physical virtual world interact to provide the necessary information and appropriate services to the end user. Cyberspace and social spaces (CPSS) [1] represent a convergence between the physical and virtual worlds, where social interactions between users are essential. In the last few years, a team from the Faculty of Mathematics and Informatics of Paisii Hilendarski Plovdiv University has been developing a reference architectural framework of Virtual Physical Space (ViPS). ViPS is a CPSS-like space that can be adapted to different domains [2].

Modeling and presenting the spatial aspects of "things" is key to creating dynamic, context-sensitive cyber-physical systems. The development of such a system is a multi-layered complex task, which includes objects and interactions, typical for different applied areas – transport, healthcare, training, etc. Therefore, the process of preliminary modeling and verification of the developed services and scenarios is essential. The article discusses an approach to Ambient-oriented modeling in ViPS [3] with complex services based on integrated domains.

The rest of the paper is organized as follows: a short review of idea for

integrated domains in ViPS is considered in the next Section 2; Section 3 presents an Ambient-oriented Modeling; in Section 4 is presented our CCA Editor. Finally, Section 4 concludes the paper.

2. Integrated domains in Virtual-Physical Space

ViPS is a reference architecture that can be adapted to develop various CPSS-type applications, such as e-learning, digitization of Bulgarian cultural and historical heritage, smart cities, precision agriculture etc. The virtual educational space VES is an integration of ViPS in the field of education [4].

ViPS includes standard components (usually interpreters) that adapt through domain-specific data structures, and new domain-specific components. The new components are stored in specialized libraries that are part of ViPS. Thus, with each adaptation, the reference ViPS architecture expands. In certain cases, it would be appropriate for ViPS to refer to more than one domain. This is how the idea of using integrated domains to provide the user with more complex services arose [5].

Consider, for example, the possibility of creating an intelligent environment in a small town. In the big city, the information that needs to be analyzed and processed is huge, and the areas of application are numerous and multi-layered. All this requires the individual domains – transport, medical care, ecology, lighting, training, etc., to be considered, modeled and implemented separately. In a small town, for obvious reasons, it is possible to create a common integrated intelligent environment in which to develop complex and interconnected and interdependent applications in different domains. For example, the "smart transport", "education", "smart healthcare" and "smart travel guide" systems can work together to solve common more complex problems. Due to the peculiarities of integrated management, it is possible to apply various optimization mechanisms, such as fast transportation of patient with a "smart ambulance" to "smart hospital" or dynamic transmission of information from the "smart bus stop" to the personal assistant of the student who is late for the exam.

The use of integrated domains in the field of education also provides many additional opportunities. For several months in FMI we have robots of third generation ROBOBO, which are designed for training in STEAM (Science – Technology – Engineering – Arts – Mathematics) laboratories [6]. The robot consists of a mobile platform (base) and a panel

on which puts a cell phone (head). The robot can perceive the environment, both with the sensors of the base and through the sensors of the mobile device, and the programming can be realized in several popular programming languages. An architecture of ATOS intelligent environment has been developed, in which the student's personal assistant communicates with Robobo's personal assistant, realizing various tasks, delivering information from ViPS digital libraries, recognizing images from a school vegetable garden, etc. In this way, the integrated domains "education" and "smart agriculture" are used [7].

Providing services to the end user from virtual physical space with integrated domains is significantly more complex and requires a preliminary process of modeling and optimization. One of the established approaches for unified modeling of objects from the physical and virtual world is Ambient-oriented modeling and in particular Calculus of Context-aware Ambients (CCA).

3. Ambient-Oriented Modeling

Ambient-oriented modeling (AOM) is a formalism through which physical and virtual objects are digitized and processed in a unified way. An ambient is an identity that has the following characteristics [8]:

- Restriction – a limited location where a calculation or action of interest takes place;
- Inclusion – one ambient can be included in the structure of other ambient. In this way, hierarchies of ambients can be created;
- Mobility – one ambient can be moved from one location to another as a whole with all its children and sub-ambient structure.

Ambients model physical, abstract, or virtual objects along with their spatial and temporal characteristics. The ambient network can be parameterized to the dynamically changing environment through the data obtained from the IoT nodes.

Depending on the location and the current state, the ambients can be static or dynamic. Static ambients have a permanent location in the physical world (eg hospitals, schools, universities, museums, etc.). According to their properties, ambients can form hierarchical structures. For example, the tourist complex "Old Town of Plovdiv" contains a whole static substructure of museums, tourist sites, monuments, exhibitions, etc. Dynamic

ambients have a variable location in the current context (eg buses, taxis, people, drones, etc.). They also have a hierarchical structure and can move out/in to other static or dynamic ambients.

The syntax and formal semantics of Calculus of Context-Aware Ambients (CCA) fully realize the mechanisms of AOM, providing much better opportunities for context-oriented modeling of the environment. In CCA, ambients are defined as identities that are used to describe an object or component process, device, location, etc. Each ambient has a name and location. There are three possible relationships between two ambients (parent, child and sibling). Ambients can exchange messages with each other. In the CCA notation, we use the symbol "::" to describe the interaction between sibling ambients; "$\uparrow$" and "$\downarrow$" are symbols of parent-child interaction; "$<,>$" means sending a message and "$(,)$" means receiving a message.

Because the CCA notation contains symbols that cannot be directly interpreted, a special ccaPL language has been developed, which is a computer-recognizable version of the CCA syntax [9]. Each line in the console interpretation of the program "$A == (X) ==> B$" means that ambient "A" sends a message "X" to ambient "B", and "Child to parent", "Parent to child" and "Sibling to Sibling" represents the relationship in the hierarchy of ambients.

The ccaPL programming language has a specific syntax [10]. The language and its interpreter were developed by a team at the Software Technology Research Laboratory (STRL) of De Montfort University. The program code is written in a plain text editor, which requires a very good knowledge of the CCA notation. This greatly complicates the process of programming the simulated test scenarios. Therefore, we decided to develop a specialized visual editor through which to implement the modeling, simulation, testing and verification of the basic scenarios for the implementation of various services in a ViPS with integrated domains.

4. CCA Editor for modeling and simulation

As it's already mentioned, the programming language ccaPL has specific syntax which is not so trivial. The programming code is written in an ordinary text editor, which requires an additional CCA knowledge from a person, who models a specific scenario. This significantly increases the complexity and complicates the programming process in different test scenarios. Because of this, we decided to develop a specialized visual editor in

order to realize the simulation, testing, and verification of the basic scenarios in various applied areas (domains). This is in order to accomplishing different services in the "smart city" and "smart education" areas.

The CCA Editor provides an opportunity to model multiple scenarios in different applied areas. This is achieved through associating of each of the ambients with specific domains and one ambient could belong to more than one domain and one domain can include more than one ambient (this is realized by "ManyToMany" relationship on the database level). Additionally, the ambient keeps reference to the messages, which it has send and/or received (Figure 1).

The CCA Editor provides several different opportunities which include: creating ambients with a choice from a predefined library in accordance to the applied area (domain); creating and exchanging messages; generation of a CCA file; opportunity for a start, test, and verification of scenarios; visualization of the exact CCA file with an option to edit and update; statistics based on the developed CCA model that help us to evaluate the complexity of a specific scenario (Figure 2).

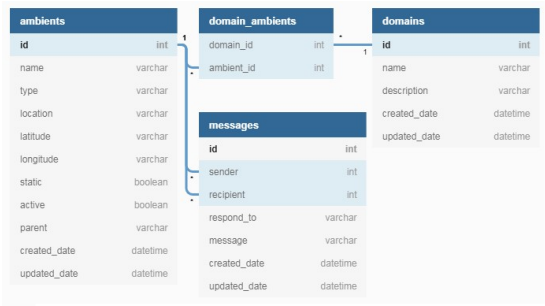

Figure 1. Database diagram of the CCA Editor

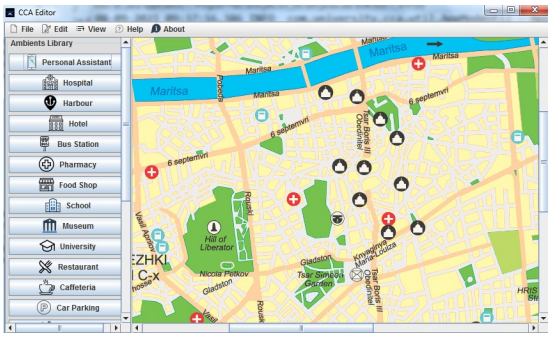

Figure 2. Main screen of the CCA Editor

Let's consider the following example scenario in order to be able to demonstrate the process of modeling, simulation, verification and testing a scenario developed into the integrated domains of "smart city" and "education" via the CCA Editor application. The student, through his personal assistant (STUDENT_PA), communicates with the educational environment of the university (PLOVDIV_UNIVERSITY) to find out if he has lectures today. After interaction with the bus station (BUS_STATION) and taxi operating service (TAXI), the personal assistant of the student is not able to find an appropriate transport and the student is not able to attend the lectures that are scheduled for today. Fortunately, the university

sends the lectures and the student receives them with the help of the virtual educational environment. Also, he receives an additional information for the exact date and time of his exams, and how he should participate in them.

The following ambients are included into the example scenario described into the next Table 1.

Ambient name	Description
STUDENT_PA	Personal assistant of the student which is an entry point of the given integrated domains
PLOVDIV_UNIVERSITY	University educational environment which is part of the "education" domain
BUS_STATION	Bus station ambient which is part of the "smart city" domain
TAXI	taxi operating service part of the "smart city" integrated domain

Table 1. Ambients in simple scenario

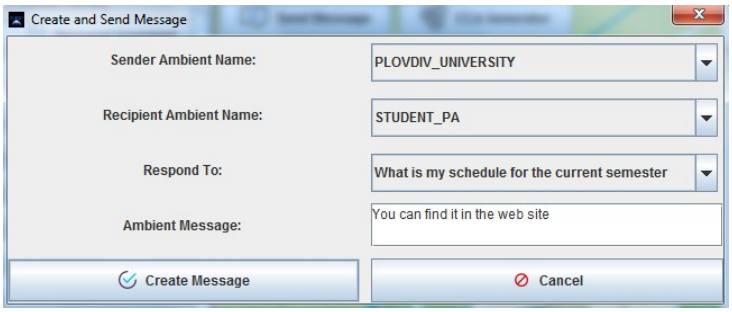

Figure 3. Communication between ambients

To be able to model, simulate, verify and test the scenario, that we already described, into the CCA Editor application, we need to follow several steps that are described below:

- Step 1. Choose the ambients from the different integrated domains. If there are no existing ambients that matched the criteria, they will be created and stored into the CCA Editor database accordingly;
- Step 2. Develop the communication between the ambients that are included into the scenario via messages (Figure 3);
- Step 3. The CCA Editor generates the ccaPL file and provides an option to the person, who models the scenario, to see and update it (Figure 4);

- Step 4. Start the already generated CCA file through the CCA interpreter in console mode (Figure 5);
- Step 5. Analyze the complexity of the ambients and the communication between them, and take some actions to be able to reduce the complexity and interaction between them;
- Step 6. Store the scenario into the library and reuse it if it's needed in the future for some reasons and purposes.

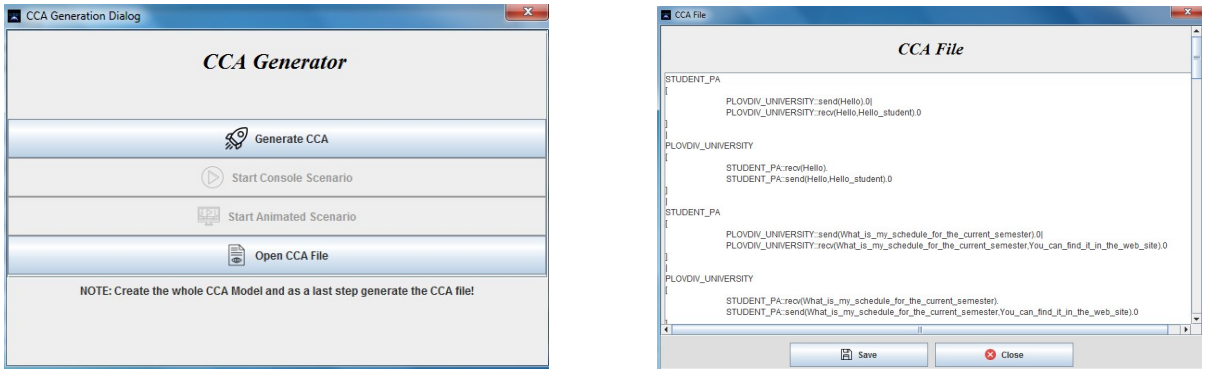

Figure 4. Generation and visualization of the CCA file

```
{Sibling to sibling: STUDENT_PA ===(Hello)===> PLOVDIV_UNIVERSITY}
{Sibling to sibling: PLOVDIV_UNIVERSITY ===(Hello,Hello_student)===> STUDENT_PA}
{Sibling to sibling: STUDENT_PA ===(What_is_my_schedule_for_the_current_semester)===> PLOVDIV_UNIVERSITY}
{Sibling to sibling: PLOVDIV_UNIVERSITY ===(What_is_my_schedule_for_the_current_semester,You_can_find_it_in_the_web_site)===> STUDENT_PA}
{Sibling to sibling: STUDENT_PA ===(Are_there_any_lectures_that_I_need_to_attend_today)===> PLOVDIV_UNIVERSITY}
{Sibling to sibling: PLOVDIV_UNIVERSITY ===(Are_there_any_lectures_that_I_need_to_attend_today,Yes_you_have)===> STUDENT_PA}
{Sibling to sibling: STUDENT_PA ===(Thank_you_for_the_information)===> PLOVDIV_UNIVERSITY}
{Sibling to sibling: PLOVDIV_UNIVERSITY ===(Thank_you_for_the_information,You_are_welcome)===> STUDENT_PA}
{Sibling to sibling: STUDENT_PA ===(Are_there_any_busses_available_soon)===> BUS_STATION}
{Sibling to sibling: BUS_STATION ===(Are_there_any_busses_available_soon,Unfortunately_no)===> STUDENT_PA}
{Sibling to sibling: STUDENT_PA ===(Are_there_any_free_taxi_cars_soon)===> TAXI}
{Sibling to sibling: TAXI ===(Are_there_any_free_taxi_cars_soon,No_the_taxi_cars_are_busy_now)===> STUDENT_PA}
{Sibling to sibling: STUDENT_PA ===(I_am_not_able_to_attend_the_lectures_today)===> PLOVDIV_UNIVERSITY}
{Sibling to sibling: PLOVDIV_UNIVERSITY ===(I_am_not_able_to_attend_the_lectures_today,Okay_it_is_not_an_issue)===> STUDENT_PA}
{Sibling to sibling: STUDENT_PA ===(Can_you_give_me_the_lectures_from_today)===> PLOVDIV_UNIVERSITY}
{Sibling to sibling: PLOVDIV_UNIVERSITY ===(Can_you_give_me_the_lectures_from_today,Sending_lectures)===> STUDENT_PA}
{Sibling to sibling: STUDENT_PA ===(Thank_you)===> PLOVDIV_UNIVERSITY}
{Sibling to sibling: PLOVDIV_UNIVERSITY ===(Thank_you,No_worries)===> STUDENT_PA}
{Sibling to sibling: STUDENT_PA ===(When_is_my_exam)===> PLOVDIV_UNIVERSITY}
{Sibling to sibling: PLOVDIV_UNIVERSITY ===(When_is_my_exam,At_the_end_of_the_semester)===> STUDENT_PA}
{Sibling to sibling: STUDENT_PA ===(Should_the_exam_be_online)===> PLOVDIV_UNIVERSITY}
{Sibling to sibling: PLOVDIV_UNIVERSITY ===(Should_the_exam_be_online,Yes_because_of_the_pandemic_the_exam_will_be_online)===> STUDENT_PA}
{Sibling to sibling: STUDENT_PA ===(Can_you_give_some_examples_for_the_exam)===> PLOVDIV_UNIVERSITY}
{Sibling to sibling: PLOVDIV_UNIVERSITY ===(Can_you_give_some_examples_for_the_exam,NO)===> STUDENT_PA}
{Sibling to sibling: STUDENT_PA ===(Okay_thank_you_for_the_whole_information_that_you_gave_me)===> PLOVDIV_UNIVERSITY}
{Sibling to sibling: PLOVDIV_UNIVERSITY ===(Okay_thank_you_for_the_whole_information_that_you_gave_me,No_worries_see_you_soon)===> STUDENT_PA}
{Sibling to sibling: STUDENT_PA ===(See_you_on_the_exam)===> PLOVDIV_UNIVERSITY}
{Sibling to sibling: PLOVDIV_UNIVERSITY ===(See_you_on_the_exam,BYE)===> STUDENT_PA}
```

Figure 5. Results of the CCA scenario in console mode

In the first version, the CCA Editor proves its effectiveness by accelerating the speed of development, as well as editing and changing the basic scenarios, which are developed. The team continues to improve the application with various features that could help the end user to model scenarios easily and without additional knowledge of the CCA and ccaPL programming language.

5. Conclusion

The report examines a context-aware modeling option known as context-aware modeling. The approach is particularly suitable for supporting the process of developing CPSS applications that provide comprehensive services to their users. The application of the approach for developing a specific model through a specially developed CCA Editor is demonstrated. The model is designed for testing and implementation in the ViPS on integrated domains.

Acknowledgments

The results published in this article are part of a study conducted with the financial support of the project FP21-FMI-002 of the Scientific Fund of the Paisii Hilendarski University of Plovdiv, Bulgaria.

References

[1] H. Yu, H. Qi, K. Li, CPSS: A study of Cyber Physical System as a Software-defined Service, *Procedia Computer Science*, (2019), Vol. **147**, 528–532, ISSN: 1877-0509.

[2] S. Stoyanov, T. Glushkova, E. Doychev, *Cyber-Physical-Social Systems and Applications – Part 1*, LAP LAMBERT Academic Publishing, (2019), ISBN: 978-620-0-31825-1.

[3] T. Glushkova, S. Stoyanov, I. Popchev, S. Cheresharov, Ambient-oriented Modelling in a Virtual Educational Space, `https://doi.org/10.7546/CRABS.2018.03.13`.

[4] S. Stoyanov, T. Glushkova, E-learning in a virtual education space, *Proc. of 4-th International Conference "Informatization of Education and E-learning Methodology: Digital Technologies in Education"*, October, (2020), Krasnoyarsk, Russia.

[5] S. Stoyanov, A. Stoyanova-Doycheva, T. Glushkova, E. Doychev, J. Todorov, A reference architecture of Internet of Things ecosystem, *Computer Sciences and Communications*, (2018), Vol. 7, No. 1, 20–28, ISSN: 1314-7846.

[6] F. Bellas et al., The Robobo Project: Bringing Educational Robotics Closer to Real-World Applications, W. Lepuschitz, M. Merdan, G. Koppensteiner, R. Balogh, D. Obdrlek (eds) Robotics in Education. RiE 2017, *Advances in Intelligent Systems and Computing*, (2018),

Vol. **630**, Springer, Cham.

[7] I. Krasteva, J. Todorov, I. Stoyanov, Intelligent agriculture in STEAM centers, *Education and Technologies*, (2021), Vol. **12**, Issue 1, 102–105, ISSN: 1314-1791.

[8] L. Cardelli, A. Gordon, Mobile Ambients, *Theoretical Computer Science*, Elsevier Science, (2000), 240177213.

[9] F. Siewe, A. Cau, H. Zedan, CCA: a Calculus of Context-aware Ambients, *Proceeding of the IEEE 23rd International Conference on Advanced Information Networking and Applications (AINA-09)*, University of Bradford, Bradford, UK, May 26–29, (2009).

[10] M. Al-Sammarraie, F. Siewe, H. Zedan, Formal Specification of an Intelligent Message Notification Service in Infostation-based mLearning System using CCA, *Proceedings of CCIT'11*, Dubai, UAE, (2011).

Todorka Glushkova[1,*], Konstantin Rusev[2], Stanimir Stoyanov[3]
[1,2,3] Paisii Hilendarski University of Plovdiv,
Faculty of Mathematics and Informatics,
236 Bulgaria Blvd., 4003 Plovdiv, Bulgaria
[*] `Corresponding author: glushkova@uni-plovdiv.bg`