AmbiNet Modeling of Spatial Aspects of Things

Stanimir Stoyanov, Todorka Glushkova, Asya Stoyanova-Doycheva, Emil Doychev

Plovdiv University “Paisii Hilendarski”, Bulgaria

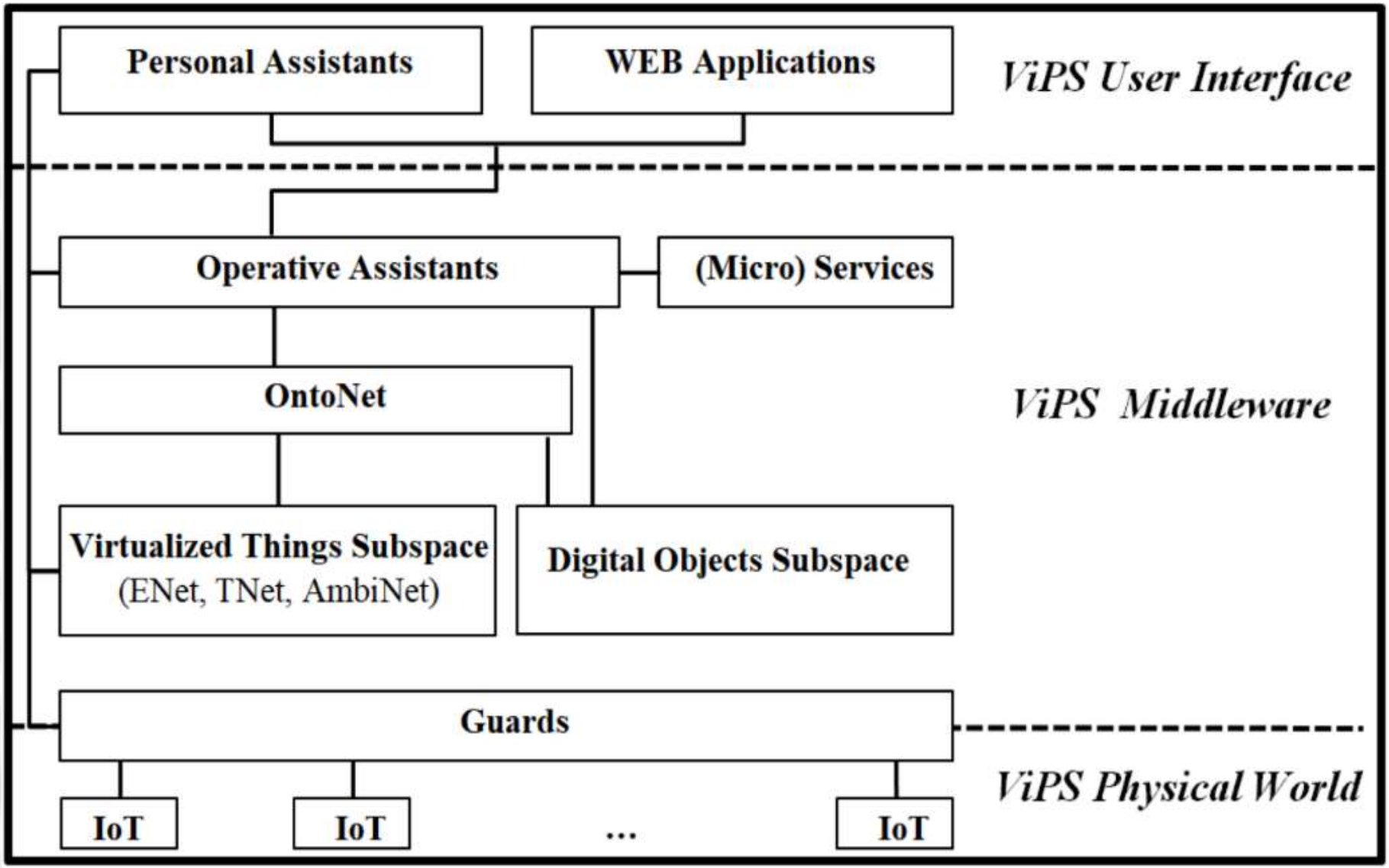
Introduction

- IoT, CPS, and CPSS are three closely related technologies, that contribute to the emergence of a new type of smart systems integrating the virtual and physical worlds.
- The reference architecture known as Virtual Physical Space (ViPS) was developed at the FMI of Plovdiv University as CPSS space.
- Three basic aspects of ViPS:
  - users that are the focus of attention;
  - virtualization of physical "things“;
  - integration of the virtual and the physical worlds.
- The article discusses the spatial aspects of the things virtualization by an ambient-oriented modeling approach.

ViPS in a Nutshell

- In particular, we have three main objectives:
  - ✓ Building a formal environment for “virtualization” of real-world “things”.
  - ✓ Creating an interface between the virtual and the real worlds.
  - ✓ A genetic personal assistant based upon a personal assistant that will help users work with this application.
- The virtualization of “things” is supported by the ViPS middleware.
- The virtualization of “things” has factors such as events, time, space, and location.

ViPS Architecture



ViPS Architecture

- The components implementing the virtualization are located in two subspaces called „Virtualized Things Subspace“ (VTS) and “Digital Objects Subspace” (DOS).
- DOS is implemented as open digital repositories, which store objects of interest to the particular field of application with their characteristics.
- VTS consists of three components:
  - ✓ AmbiNet- spatial characteristics of the "things"
  - ✓ ENet - models various types of events
  - ✓ TNet - the temporal aspects of things.

AmbiNet

- AmbiNet is a ViPS component for modeling the spatial aspects of "things". This component is implemented as an extension of the original ccaPL environment.
- The ccaPL environment, based on the CCA syntax, is implemented as a Java application with two components:
  - ✓ Development tools- ccaEditor and ccaParser.
    - ccaEditor is used to encode the model as a ccaPL program;
    - ccaParser operates as a syntax checker.
  - ✓ The ccaPL run-time - ccaInterpreter, ccaConsole, and ccaAnimator.

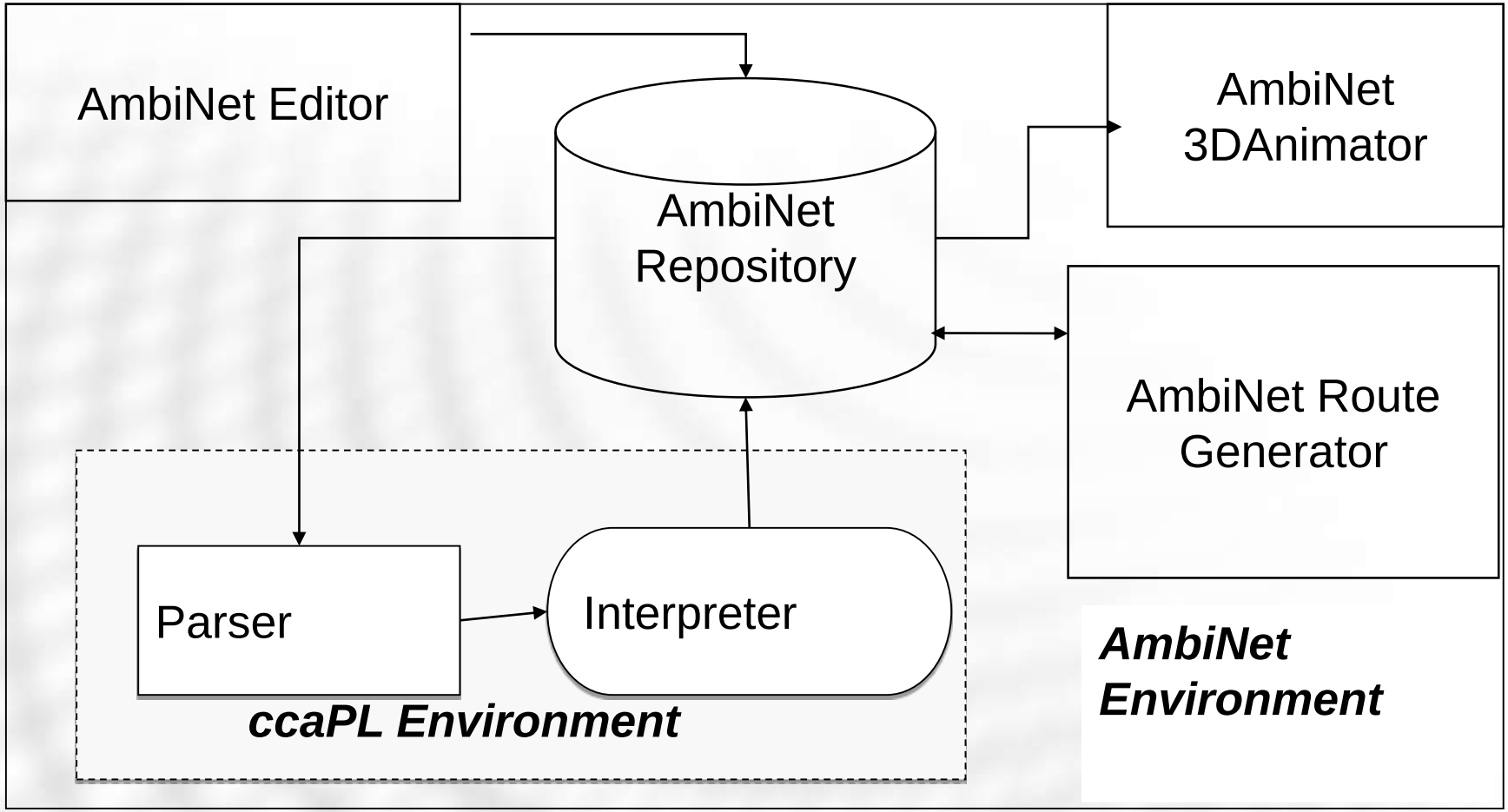
AmbiNet

The original ccaPL environment has some drawbacks that make it difficult to use directly for various domains.

New components of the reengineering AmbiNet environment:

- ✓ **AmbiNet 3D Animator**. In AmbiNet, the results are explicitly stored in an intermediate file. This file, along with the graphical pattern from the AmbiNet Repository, is used to present the results in 3D mode.
- ✓ **AmbiNet Route Generator**. The results of the interpretation provide potentially usable “things” modeled as ambients. They are subject to additional checks in order to be included in the appropriate route generation.
- ✓ **AmbiNet Repository**. All components of the AmbiNet environment are synchronized through this repository. It provides the needed data for the AmbiNet components.

AmbiNet Architecture



An Example - Smart City

- The development of "Smart city" system is a complex task that requires an in-depth process of preliminary modeling.
- Such a model requires virtualization of different types of "things" from the surrounding physical and virtual worlds - transport, public places, facilities for people with disabilities, devices for monitoring the air, water, etc.
- Through such a model we can study in advance the behaviors of some services that a Smart City can provide to its users.
- Each user has his/her own personal assistant (PA) to interact with the Smart City system.

An Example

- A tourist with an intelligent wheelchair and respiratory disability is at his/her hotel in a "smart city" and he/she has a personal assistant PA to interact with the Smart City. The tourist wishes to visit a sightseeing spot using a taxi.
- There is a factory not far from the city that sometimes emits suffocating gas and pollutes parts of the city; the tourist should avoid the polluted areas (if any).
- The city is equipped with multiple sensors that dynamically transmit data as well as activity information of devices important for the wheelchair - ramps, elevators, automatic doors, etc.

An Example

- This scenario has two parts:
  - ✓ finding the appropriate route to visit the sightseeing spot by taxi;
  - ✓ providing the route to transport the user to the local hospital.
- Communication starts between the intelligent components of the Smart City for the rapid movement of the patient to the local hospital.
- The transportation will be made with a “smart” ambulance, which will interact dynamically with other objects such as “smart” traffic lights and emergency teams in the hospital to provide adequate emergency medical care.

Ambients in AmbiNet

- The AmbiNet is used to model these scenarios.
- We can look at two sets of ambients: abstract and physical.
- According to the type of their physical location, the second group is divided into two types: static and dynamic.
- Static ambients have constant location in the physical world (hotels, hospitals, museums)
- Dynamic ambients have a variable location in the physical world (buses, ambulances, elevators).
- Each ambient can contain a hierarchical structure of other ambients
- We will present each ambient as:

a = <name, location, type, parent, P (a), attr (a)>

Ambients in the simple scenario

- The Physical static ambients are: Tourist attraction, Hospital, and Traffic Light
- The Physical dynamic ambients are: Wheelchair, Taxi, Ambulance.
- The abstract ambients are: Personal Assistant (PA), AmbiNet, Route Generator and Guards.

The processes of AmbiNet ambient are:

$$
\begin{aligned}
&PA :: (PA_i, loc, visitFort).Fort ::< PAi, visitFort > .0 \mid \\
&Fort :: (PA_i, FortMuseumWorkingTime). \\
&Taxi ::< PA_i, TravelByTaxi > .0 \mid Taxi :: (PAi, hasTaxi). \\
&PA ::< hasTaxi, openFortMuseum > .0 \mid \\
&PA :: (PA_i, loc, getRouteToFort). \\
&Guards ::< PA_i, getActiveZ, getAllowedZ > .0 \mid \\
&Guards :: (PA_i, ListActiveZ, ListAllowedZ). \\
&RG \downarrow < PA_i, loc, ListActiveZ, ListAllowedZ, getRoutes > .0 \mid \\
&RG \downarrow (PA_i, ListRoutes).PA ::< Route > .0 \mid \\
&RG \downarrow < PA_i, loc, loc\_Hosp, getRoute > .RG \downarrow (PAi, Route).0 \mid \\
&Guards :: (PA_i, loc, TrLightStatus, AmbulStatus). \\
&Guards ::< PA_i, Route > .0
\end{aligned}
$$

Ambients in the simple scenario

- The Guards play a central role in the interaction between the virtual and the physical worlds.
- The Physical ambients Hosp, Traffic lights and Taxi communicate with the virtual ambients through Guards.

The processes of Guard ambient are:

$$
\begin{pmatrix}
AmbiNet :: (PA_i, getActiveZ, getAllowedZ). \\
AmbiNet ::< PA_i, ListActiveZ, ListAllowedZ > .0 \mid \\
PA :: (PA_i, location, TransportToHospital). \\
Hosp ::< PA_i, location, needAmbul > . \\
Hosp :: (PA_i, AmbulStatus). \\
AmbiNet ::< PA_i, location, AmbulStatus > .0 \mid \\
AmbiNet :: (PA_i, Route).PA ::< Route > . \\
Hosp ::< PA_i, Route > .0
\end{pmatrix}
$$

AmbiNet in action

AmbiNet Editor uses templates of the physical city from ***AmbiNet Repository***.

AmbiNet Route Generator component generates the route.

The interactions in current scenario are presented as a ccaPL-program and the ***ccaInterpreter*** and ***ccaAnimator*** simulate the described model.

AmbiNet in action

- The communication between ambients in current scenario, gives us preliminary information about the workflows and the workload of the individual components in the system.
- The analysis of these interactions allows us to determine the load of the individual ambients, which allows us to correct the simulated scenario at the stage of modeling and prototyping.
- We save the processed and optimized model in the AmbiNet Repository, from where it can be retrieved for use in the actual execution process.

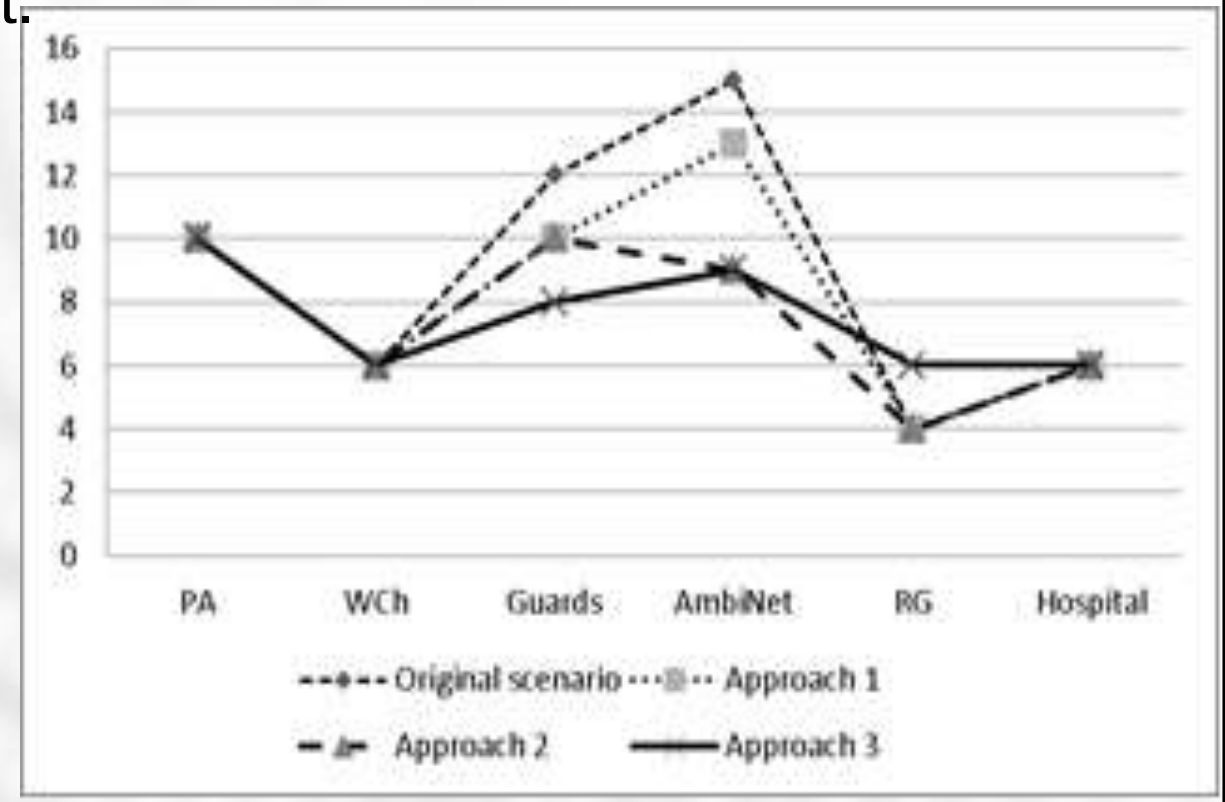
Different Approaches

- For this scenario, we notice that the PA, the Guards and the AmbiNet ambients are the busiest.
- The first one realizes 10, the second – 12, and the third one – 15 interactions.
- While this is largely justified for the PA, which provides the customer connection, for the Guards and AmbiNet we can say that the workload is too high and the model needs further corrections.
- There are several approaches to achieve this:
  - ✓ Approach 1: by transferring some of the communication to other ambients;
  - ✓ Approach 2: by using the ambients mobility feature.
  - ✓ Approach 3: by implementing a combined approach.

- In the example considered, it is possible to start a parallel process between the Guards ambient with IoT ambients for up-to-date information from the real world and periodically send this information to AmbiNet.
- This would reduce the number of requests from AmbiNet, but it would increase the communication between the Guards and IoTn many times, as this communication would be implemented periodically.
- The problem can be solved by using Edge and Fog Computing.

Since ambients can change their locations, we notice that in the communication with the Guards to determine the activity of individual nodes (IoTn) in the route generation process, the RouteGenerator (RG) ambient may exit temporarily beyond the AmbiNet parent ambient.

In this way, the interactions of RG and the Guards with AmbiNet decrease. Combining the two approaches results in even better results, burdening the participating ambients more evenly.



Conclusions

- Preliminary AOM-modeling in Smart City makes it possible to determine the feasibility of the created workflow, as well as to optimize the workload of individual components. This determines the need to develop a specialized analyzer and optimizer tool for AmbiNet.
- A prototype of the environment is fully implemented in Java. The personal assistant PA, integrated in AmbiNet, is implemented in the Jason development environment.
- The AmbiNet prototype is in the testing stage. To prove the prototype's performance, the authors have prepared a test suite containing scripts from various application areas. The example discussed in the paper is related to modeling of services in a smart city.

Thank you!

Questions?

glushkova@uni-plovdiv.bg