Journal of Ambient Intelligence and Humanized Computing

Modelling Routine Interactions with Intelligent Environments

--Manuscript Draft--

Manuscript Number:	
Full Title:	Modelling Routine Interactions with Intelligent Environments
Article Type:	S.I. : AmI for Smart Environments and Smart Cities
Keywords:	scenarios; routine procedures; intelligent environment; ambient intelligence; environment modelling
Corresponding Author:	Peter Mikulecky, PhD Univerzita Hradec Kralove Hradec Kralove, CZECH REPUBLIC
Corresponding Author Secondary Information:	
Corresponding Author's Institution:	Univerzita Hradec Kralove
Corresponding Author's Secondary Institution:	
First Author:	Peter Mikulecky, PhD
First Author Secondary Information:	
Order of Authors:	Peter Mikulecky, PhD
	Petr Tucnik, PhD
	Tomas Nachazel
	Petr Blecha
Order of Authors Secondary Information:	
Funding Information:	Grantová Agentura České Republiky (GA15-11724S) — Prof. Dr. Peter Mikulecky
Abstract:	This paper describes the use of scenarios to model interactions based on the routine procedures of inhabitants (users) with intelligent environments. For a comprehensible and illustrative description of various features of use scenarios, a specific case study of a client of a spa resort is introduced and a technical background description is provided. The model of an intelligent environment presented here was created based on the real-world spa resort of Trencianske Teplice (situated in the western Slovakian Republic), and includes both interior and exterior components over a limited geographical area. Attention is focused on the creation of various functionalities tailored to the needs of the user (here, a client of the spa resort) according to his/her daily routines and expectations of available services. The scenario examined here shows positive potential for improving the analytical and design phases of development of an intelligent environment.

Modelling Routine Interactions with Intelligent Environments

Petr Tucnik, Tomas Nachazel, Petr Blecha, Peter Mikulecky

Faculty of Informatics and Management,
University of Hradec Kralove,
Hradec Kralove, Czech Republic
petr.tucnik@uhk.cz

Abstract. This paper describes the use of scenarios to model interactions based on the routine procedures of inhabitants (users) with intelligent environments. For a comprehensible and illustrative description of various features of use scenarios, a specific case study of a client of a spa resort is introduced and a technical background description is provided. The model of an intelligent environment presented here was created based on the real-world spa resort of Trencianske Teplice (situated in the western Slovakian Republic), and includes both interior and exterior components over a limited geographical area. Attention is focused on the creation of various functionalities tailored to the needs of the user (here, a client of the spa resort) according to his/her daily routines and expectations of available services. The scenario examined here shows positive potential for improving the analytical and design phases of development of an intelligent environment.

Keywords: scenarios, routine procedures, intelligent environment, ambient intelligence, environment modelling.

1 Introduction

When faced with the task of designing an intelligent environment, there are numerous possible ways to approach this, with many optional functionalities that may or may not be included in the design. To make a qualified (or at least more accurate) decision about what to include, it may prove useful to have information about the key functionalities which are expected to be frequently used by an environment`s inhabitants. These functionalities originate from the routine procedures that users undergo in their daily lives. One possible approach that will be used here is the use of scenarios. Scenarios are already a well-established concept in the domain of analysis and modelling; they are commonly used in the well-known Unified Modelling Language (UML) as part of use case diagrams, but may also represent a less formal description in natural language or more rigorous diagrams with pre-established syntax.

Routine behaviour influences many aspects of our lives. According to Hodgson [1], *routines* are defined as frequent actions that people perform within different situations that are the cause of these actions. Routines are a type of purposeful behaviour composed of goal-directed actions, which people acquire, learn and develop through repeated practice [2]. As such, good routines enable the predictable and efficient completion of frequent, repetitive tasks and activities. Routines can describe a number of behavioural patterns, for example sleeping or exercising patterns, or even low-level tasks, such as how drivers operate their cars in specific situations. Routines, like most other kinds of human behaviours, are not fixed but may be varied and adapted based on feedback and preferences [2].

In this paper, we show how to use scenarios to model the routine behaviour of people in an intelligent environment, with a special focus on routine interactions with the environment. Our model of an intelligent environment is demonstrated in the case of a real spa resort known as Trencianske Teplice, a famous resort in Western Slovakia.

2 State of the Art and Relevant Works

According to [3], intelligent environments based on the ambient assisted living paradigm seem to be a key resource in assisting people in their daily life. The populations of most developed countries are aging, and their structure is changing. Consequently, many people live alone, especially seniors, and the ubiquitous health monitoring of these people has become a crucial issue not only in terms of prevention or emergency detection, but also in terms of assisting them whenever it seems to be necessary [4].

The sophisticated use of sensors and computing devices enables intelligent environments to monitor the activity of inhabitants, and to plan suitable interventions related to their activities and the analysis of routines or stereotypical behaviour of the people within these environments. However, real intelligent environments are expensive to build and maintain. If there is a need to test a new sensor within an environment, it is typically rather difficult to reconfigure the environment for a new series of experiments. As Helal et al. stressed in [5], the existence of an intelligent environment does not guarantee the generation of the necessary datasets, since recruiting humans for experiments is a difficult process. It is clear that these issues slow down the research process, and hinder the ability to conduct meaningful experiments in intelligent environments.

Many researchers have therefore proposed the development of various approaches to the simulation of both intelligent environments and the possibility of mutual interactions between this environment and its users (see e.g. [6], [7], [8] or [9]). In general, when experiments are expensive, time-consuming or ethically problematic, a simulation can be a suitable tool for the first phase of research and development [4].

In order to run a successful multi-agent simulation (in which agents represent various parts of the intelligent environment emulated in the model), it is necessary to provide appropriate scenarios for agents. According to [10], a description of the social interactions between agents and humans is essential if we want to conduct large-scale multi-agent simulations. Thus, scenarios for simulation should be written by experts in a given application domain. These scenarios capture characteristic, usually repeated, situations and serve either as an example of how to behave according to social or other norms or how to support the decision-making processes of intelligent components by highlighting an appropriate context. A system is also more precise in activation of proper components, enabling it to act unobtrusively and without unnecessary interaction with its users; this is generally expected of such a system.

Scenarios for multi-agent simulations in an intelligent environment are descriptions of more or less complex behaviours, made up of sequences of events or elementary actions that follow on from each other. The term 'activity' is frequently used to denote an action or an elementary event that is a component of some scenario. Obviously, the duration of each activity and the delay between two activities are of importance for activity recognition [11]. Scenarios describe how the system components, the environment and the users work concurrently and interact in order to provide system-level functionality. Each scenario is a partial description that, when combined with the other possible scenarios, contributes to a description of the overall system [12].

Various approaches are used to describe scenarios, for example possibilistic logic, which is used for modelling temporal scenarios [11], or situation theory, as used by Pinheiro [13]. An important and exhaustive survey of the various tools and platforms for general agent-based modelling has been carried out by Nikolai and Madey [14]. In the following, we describe several modelling and simulation platforms that are closely related to the topic of this paper.

An interesting attempt at defining a new language called SDLMAS to describe scenarios in multi-agent systems, independently of the target platform and implementation language, was done by Cavrak, Stranjak, and Zagar [15]. It adopted an interaction-centric approach with a focus on message flow between agents, providing an intuitive scenario description. According to [15], the SDLMAS platform was created in order to equip designers with the facility to describe negotiation scenarios and to allow developers to concentrate solely on an agent's logic, by generating the necessary helper classes to support message passing, execution of scenario actions and the invocation of certain procedures. The development cycle was shortened by several iterations, and significant increases in the reliability and stability of the system were found after deployment.

Many authors have already tackled the problem of creating a suitable language or approach for scenario descriptions. Several approaches have been based on AgentUML [16], Petri nets [17] or state-chart diagrams. As Paurobally et al. [18] stressed, Petri nets are not very suitable for describing interactions due to their lack of clarity and scalability. State-charts offer a clearer representation of interaction protocols, but these still lack a clear definition of the relationship between protocol execution and an agent's operational logic.

The development of AgentUML (AUML) [16], [19] was another attempt to represent agents' interactions and roles in a standardised way. The UML standard was used here as a basis for a paradigm shift from object-oriented to agent-based concepts, and to enable a standardised notation for the analysis, design and implementation of agent systems. The UML class diagram was amended to include concepts of roles and behaviours, while the sequence diagram was extended with the specificities of agent interactions. One of the advantages of AUML is that it offers a visual representation of agent dialogues; however, AUML lacks sufficient capability to represent the agents' states, giving rise to an inability to define the conditions under which messages can be received or sent by an agent [15].

General-purpose large-scale agent-based modelling software applications, such as Repast [20] and MASON [21], provide interactive tools for the stages of agent modelling, from model specification to the visualisation of results. Communication between agents is achieved through peer-to-peer mechanisms that allow the direct invocation of methods between them [3]. More detailed recommendations for the development of agent-based models (including a thorough description of all the stages of model development mentioned above) can be found in papers by Kaminsky and Szufel [22] and Ormerod and Roosevelt [23].

Since simulated intelligent environments offer many positive features, for example the generation of large datasets, total control of the environment and sensor layouts, cost-effective experiments and the ability to define very specific experimental conditions [3], a number of original simulators have been developed that aim to simulate relatively complex scenarios in intelligent environments. An important review by Synnott et al. [24] divides simulators into two main groups: model-based simulators and interactive simulators. Model-based approaches use activity models, obtained in diverse ways, to create synthetic datasets that store information about the activation of sensors during the execution of activities. Interactive approaches are based on the assumption that a human user can interact with the virtual environments and sensors set up by the simulation. In this case, the state of the virtual environment and sensors changes depending on the actions executed by the user, and the related information is stored in a synthetic dataset. In brief, model-based approaches facilitate the generation of data based on activity models, while interactive approaches are based on the use of virtual environments and virtual sensors, which respond to user interactions [24].

Helal et al. [25] demonstrated an example of a model-based approach in their PerSim simulator. PerSim was developed to facilitate the synthesis of data for the testing of activity recognition research. The simulator allows users to define activities by specifying the sensors involved in each activity, the order of activation of sensors, the maximum and minimum typical sensor values and the duration of the activity. Based on these parameters, a list of sensor data can be generated using Sensory Dataset Description Language [24]. This synthesised dataset can contain data describing the results of performing individual activities, or the results for an entire space, including those sensors that are not triggered directly by an activity, such as temperature sensors [25].

Kormányos and Pataki [4] have developed a simulator that is capable of modelling the activity of a single inhabitant within an intelligent environment. This approach facilitates the modelling of individual behaviour profiles, such as the typical period of sleep and changes in the current state, such as thirst and tiredness. This approach is capable of outputting data from simulated motion sensors, RFIDs and water consumption. A change in the current state, such as thirst, influences the likelihood of an activity occurring, for example drinking.

Kamara-Esteban et al. [3] recently presented the MASSHA simulator. This is an agent-based simulator for human activities in sensorised spaces; unlike many previous approaches, it is based on environmental multi-agent theories to model and simulate intelligent environments, inhabitants and their interactions. Multi-agent environments allow for bottom-up modelling, in which all agents collaborate and compete against each other while interacting with the environment. A model is defined and a simulation run in which the results vary depending on both the characteristics of the agents and their decision processes. The advantage of this approach is the possibility of parameterising different scenarios and evaluating how the agents' behaviour and interactions are affected in each case. The scope of MASSHA extends far beyond the replication of human activities, and its functionality can be applied to other scenarios.

In MASSHA, the communication between agents and the environment follows a publish–subscribe mechanism; an agent in MASSHA can subscribe to certain types of events, whether from the environment or from other agents. When an event takes place, all subscribers receive an automatic notification. This approach allows the communication to be very independent, providing a robust and scalable context for communication and interaction regardless of the complexity of the human activity model [3].

Finally, Buchmayr et al. [26] introduced a simulator for the generation and visualisation of sensor data within intelligent environments. This simulator displays a virtual environment using a 2D floor plan layout, and facilitates user interaction with virtual sensors via mouse clicks within the floor plan, which generate sensor data output to a log file. The simulator supports the use of simple sensors such as binary, contact and temperature sensors that emit a signal upon activation, and complex sensors such as motion and pressure sensors that emit a signal periodically after activation. The simulator also supports the generation of random data to simulate sensor faults. The addition of sensors to the 2D floor plan is supported through a drag-and-drop mechanism; however, the creation of new sensor types requires the development of data models, parsers and filters for each sensor, meaning that this functionality is less accessible for non-technical users. This study did not allow interaction between avatars, and gave limited details about the processes of creating a virtual environment or visualizing data [26].

3 Simulation Model of User-Environment Interactions

3.1 Motivation

The simulation model, described in detail later in this chapter, involves the use of a scenario-based approach to modelling the interactions of a user (inhabiting, moving and interacting within given geographical area) with an intelligent environment. For a more complete modelling of the problem, a combination of internal and external perspectives is used, and it is assumed that the user moves freely within the area shown in **Fig. 1**.

The situational context is as follows. The user is a client of the Trencianske Teplice spa resort, where he/she resides for several weeks (typically a month, but no less than two weeks), and during this time undergoes various treatment procedures. From an intelligent environment design perspective, several key factors of this situational setting are important:

- The user (client) has enough time in the environment to develop a certain regularity in his/her activities.
- This daily regime is strongly dependent on the treatment procedures (undergone at specific times) and the daily cycle of hotel services (e.g. meals are served at specific times of the day).
- The user's movement is conveniently (and consensually) limited to a certain geographical area. In the majority of cases, a European (Slovakian) spa resort resembles a town, with all its services in one place, and apart from touristic reasons, clients generally remain within its limits for the whole duration of their stay. Spa resorts include everything necessary for a pleasant stay: hotels, restaurants, treatment facilities, sport and entertainment facilities, and large parkland areas for walks, using paths of various lengths. Since the area is geographically limited, the components of an intelligent environment can be placed as needed to provide the respective functionalities and services.
- During the stay, there are fewer ad hoc activities than in a normal daily regime (e.g. for a working person). This makes this setting more suitable for the application and study of scenarios.

Our goal is to study the behavioural patterns of the user during his/her stay. Our attention is particularly focused on which facilities are used (this is partially dependent on the individual health requirements of the particular client and the respective treatment procedures that are undertaken) and which functionalities are activated by the client. The client's activities and his/her interactions with the environment are monitored and stored in the activity log, allowing subsequent analysis and enabling the design of the intelligent environment to be tailored specifically to the spa resort users' needs. Although the model scenario used in the following sections of this paper deal with a single specific user, it is expected that these data can also be obtained from other users in order to obtain objective requirements for the system`s functionalities. This approach can be used either before the components of the intelligent environment are implemented or after their installation. In the latter case, the system designer is able to verify that the design is correct and facilities are being used purposefully and efficiently.

3.2 Environment and Central Home System

Due to its convenient location and the availability of its spa services, the city of Trencianske Teplice (located in the western part of the Slovak Republic) was chosen as a suitable template for the virtual environment model. Buildings, roads, and available services were created in the agent model based on the map shown in **Fig. 1**.

In the model, the intelligent system monitors the behaviour of a single user, an agent entitled Person. This agent represents an elderly person who is a client of the Trencianske Teplice spa resort, and who uses a number of its services. He/she is able to use all services throughout the resort depending on his/her schedule. He/she can relax in the swimming pool, which is near to his/her apartment, or try any spa procedure such as acupuncture, mud wraps, or electrotherapy. There are several places serving food; alternatively, the user can cook a meal in his/her apartment.

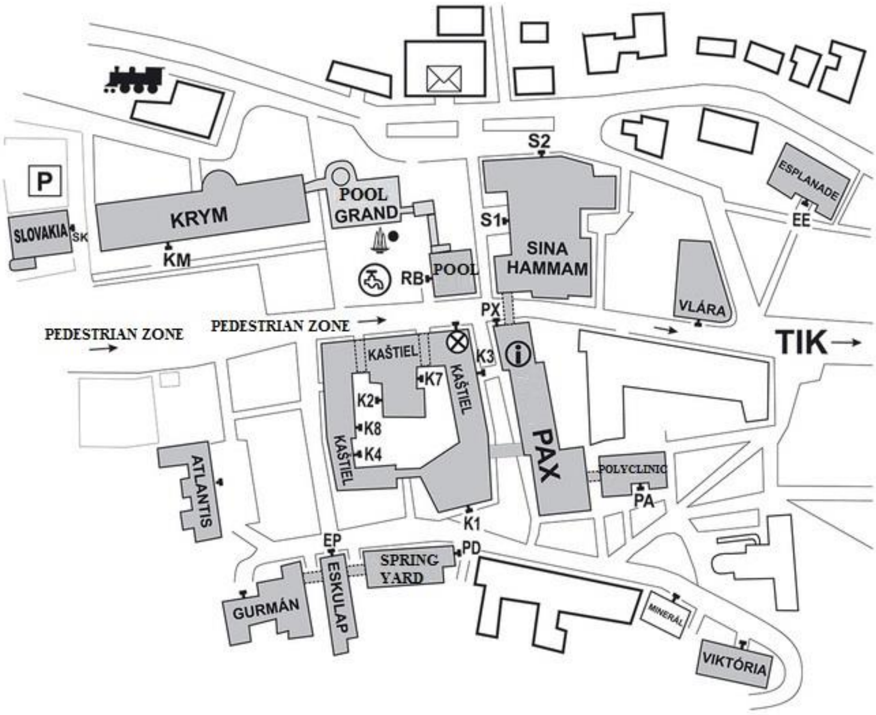

Fig. 1. Map of the town of Trencianske Teplice

The user's apartment is located in the hotel near to the train station. The user's apartment is equipped as a standard city flat (the template is based on the real flat, as can be seen from the illustration in **Fig. 2**), and is equipped with kitchen appliances and standard sanitary facilities. The apartment offers various functions of the smart technology, which are managed by a component called the Central Home System (CHS). These functions include motion sensors, appliance trigger detectors, and pressure sensors. The CHS unit processes the inputs from all sensors in this interior space.

The CHS also manages actuators, which may affect the functionality of the environment; for example, the CHS may turn off the TV if it decides that the user is not able to watch it (e.g. having moved into another room for a certain period of time or having left the apartment).

The user's movements are captured by the motion sensors located in all rooms of the apartment. Information about the position (which is combined with information from the pressure sensors) helps the CHS decide if an action needs to be performed.

Sensors within appliances (such as the stove, sink or TV) detect whether or not the user is using the appliance. There is a risk that if the user forgets to turn off the appliance (e.g. stove) and goes to sleep, the appliance will still consume energy or an accident may occur. The CHS tries to prevent this situation by deciding based on the collected data whether it should intervene or wait for the user's intervention.

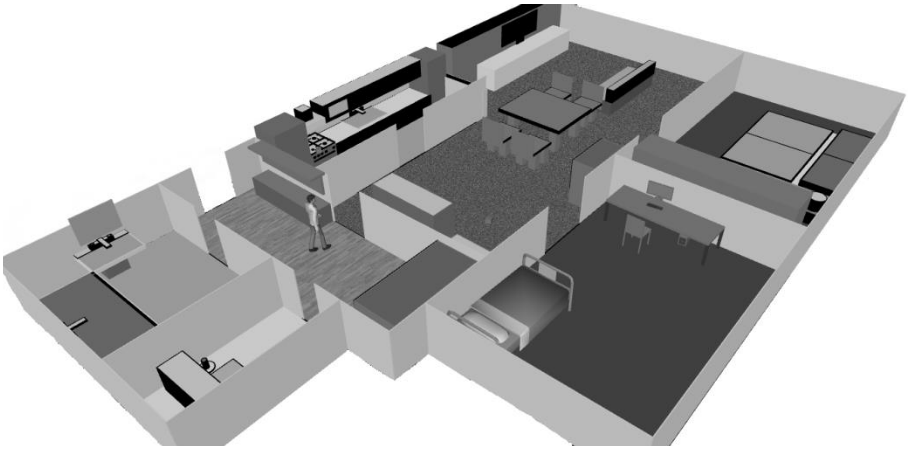

Fig. 2. Illustration of the apartment in Anylogic

The user's behaviour, as described by data about the client, data collected from sensors, and data about actuator activity, is tracked in the system and stored in an external file (log) to analyse the entire simulation run. In the each log record, the following data are stored:

- Real time-stamp
- Simulation time-stamp
- Originator (e.g. motion sensor in the kitchen)
- Activity (e.g. user is captured by kitchen motion sensor)

These logs are used as outputs to subsequently verify and analyse the behaviour of all simulation participants (agents). The general types of agents in the model are listed in **Table 1** below.

Table 1. Descriptions of agents in the spa resort model

Agent	Description
Person	The user of the services of the spa resort
Sensor	A device to collect data with a particular specialisation
Actuator	A device to perform actions managed by the CHS

Agent	Description
CHS	The system managing sensors and actuators, and deciding on actions.

When starting the simulation of the Virtual Spa Environment, several functions are performed in which the agents are initialised. Information about the agents (location, type, dependencies) is loaded from the SQL database, allowing easy reconfiguration of the actual state of the environment.

In the beginning, the Person agent loads the scenario and starts making decisions about its activity. It chooses a current activity based on the planned schedule and the unexpected immediate needs of the person (such as thirst or fatigue, which temporarily take priority). It considers activities that are planned for the current period, expected very soon, or existing but delayed. The schedule status and the importance of the activity is then compared with the person's simulated needs such as hunger, thirst, or hygiene. The simulated user then walks through the environment, trying to complete scheduled tasks and meet his/her needs, while the sensors of the intelligent system keep track of his/her behaviour.

Once a sensor is created, it executes its behavioural procedure. A motion sensor checks whether the user is present within a reachable distance. If this condition is met, a message is sent to the CHS that the motion of the user was captured. The behaviour of a motion sensor agent is shown in **Fig. 3**.

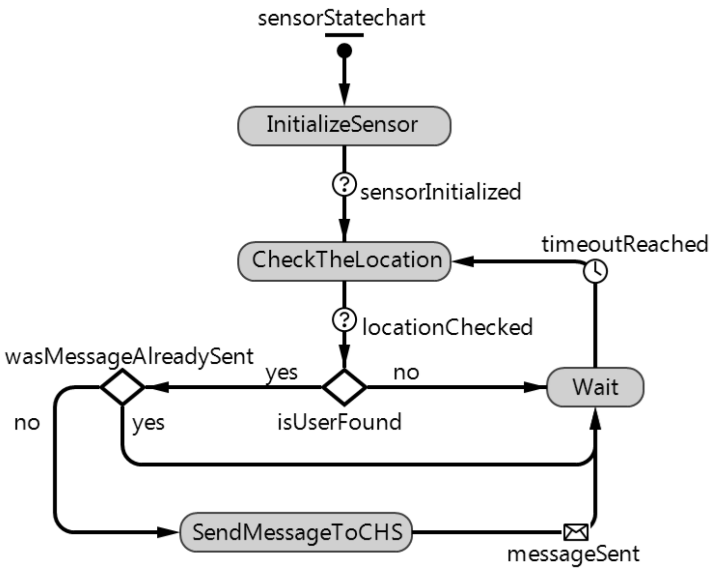

Fig. 3. State chart representation of the behaviour of the motion sensor detector

Fig. 3 provides an example of sensor implementation. Sensors vary in terms of their details (depending on what is being measured, what specific information is sent, etc.), but the governing principle of their function is the same. A list of steps used by sensors with a description of their meaning is given in **Table 2**.

The sensor retains information about the last measurement. If the previous measurement indicated that the user was captured within a specific location, no other message needs to be sent to the CHS, since the system already has the record.

Table 2. Description of the steps of the behavior state chart for a motion sensor

Step	Description
InitialiseSensor	A state that occurs only once within the sensor life cycle; state initialisation is performed
CheckTheLocation	A state in which the sensor checks whether the user is within the measuring range of the sensor.
isUserFound	Decision on whether the user was captured
wasMessageAlreadySent	Decision on whether a message has already been sent to the CHS
SendMessageToCHS	Sending of a message with information about the user's location
Wait	Waiting state (configurable value, set to one second by default)

The CHS checks the database where the sender (sensor) is located, and updates its information about the position of the user.

Each appliance is equipped with its own sensor, which checks whether the appliance is turned on or off. These sensors have a behaviour that is similar to that of the motion detector sensors, that is, checking the actual state (of the appliance) and sending the results to the CHS.

The CHS collects information from the sensors and sends messages to the actuators to regulate the state of the environment based on the current situation.

The following is an example scenario for a user's behaviour:

- User is located in the bedroom
- User stands up from the bed and moves across the living room to the kitchen
- User turns on the stove to make dinner
- User goes into the living room to eat dinner, forgetting to turn off the stove

The whole sequence of activities, captured via the sensors and actuators, can be logged in terms of the information received, as shown in the following example of log records:

20:00:00, 11:25:00, PressureSensor_Bedroom, User is captured on the bed.
20:00:00, 11:25:05, MotionSensor_Bedroom, User is captured in the bedroom.
20:00:01, 11:45:02, MotionSensor_LivingRoom, User is captured in the living room.
20:00:01, 11:45:08, MotionSensor_Kitchen, User is captured in the kitchen.
20:00:01, 11:45:15, ApplianceSensor_Stove, Stove was turned on by user.
20:00:05, 12:22:36, MotionSensor_LivingRoom, User is captured in the living room.
20:00:05, 12:25:00, CHS detects user in a different location. Stove is turned off by the CHS.

These logs provide information from the sensors and the CHS about the captured behaviour and the CHS intervention. The timespan of the simulation can be modelled within a shorter real time, and this example takes five seconds; the computational complexity of the model allows the processing of months of simulation time within a few seconds if a larger amount of output data is needed.

3.3 Implementation of Scenarios

Scenarios use various formats and representations of time or continuity, depending on the input format used [10]. However, any scenario for the model must be transferred into a machine-readable format such as a CSV file, XML structure, or table. This paper describes an implementation based on tables and uses a database to store the data; these tables are also readable by humans, which is one of the requirements of the model design and increases the utility value of the obtained data.

This representation involves two tables, containing all of the necessary information about a scenario. The first (see **Table 3**) stores the schedule, with data exclusive to each occurrence of an activity in the schedule. Two items are required: the action itself, and the time it starts. These are complemented by the probability parameter, which is used during decision-making and reflects the importance of the activity. If the agent is under time pressure, and has less time than required by a scheduled activity, it skips or postpones less important activities.

Table 3. Example of a table with a schedule

Time	Activity	Probability
6:00	bathroom	0.95
6:15	medicine	1
6:30	exercise	0.7

The second table contains general information about all possible activities, regardless of when or how often they occur in the schedule (see **Table 4**). Both tables are unlimited in both dimensions; rows can be added for new activities or columns for additional parameters about activities. The second table holds information about the duration, place and effects of an activity. The parameters of the activities used in the model are described in **Table 5**.

Table 4. Example of a table with additional data about activities

Activity	Duration (min)	Max multiplier	Place	Needs to be completed	Required activities	Uses	Consumes	Conditions	Effects

bathroom	10	1.3	bathroom	true		sink			hygiene += 0.2
medicine	4	1	bathroom	true					
exercise	15	1.5	livingRoom	false				fatigue < 0.8	hygiene -= 0.05/min
drink	1	1		true	get_drink		drink		thirst -= 0.8

Table 5. Description of parameters of activities

Duration	How long it will take until the activity is finished (minimum time)
Max multiplier	How many times longer it may take
Place	Where the activity takes a place
Needs to be completed	If true, the effect of the activity is applied only after it is ended. Otherwise, the activity affects the agent continuously
Required activity	When the activity cannot be performed, the agent executes a required activity instead (to collect a consumable item or prepare conditions for the activity).
Uses	Any furniture or device used during the activity
Consumes	Any item in agent's inventory which will be consumed by the activity
Conditions	Which conditions need to be met to proceed with the activity
Effects	How the activity affects the person

To be able to work with a scenario in the model, the agent loads it from the database and stores it as a schedule of activities. Each activity stores information about where and when it is supposed to happen, any object it consumes, any equipment it uses, and so on. Since the model is non-deterministic and possible unexpected events can occur, agents also have to deal with the possibility of delaying or even skipping a planned activity. To handle these situations, the schedule has a buffer where it stores activities which are about to begin or are already delayed.

Each activity has a fuzzy value determining its probability or importance. If an agent does not manage to complete the schedule and the buffer contains more activities, then these values and the original starting times determine which activities should be prioritised.

Activities can be split into two groups in terms of their effectiveness. In the first group, activities have a duration that is objective and has an expected length, and their effect is imposed only when they are fully completed. If the activity is interrupted, it is necessary to start from the beginning or to continue some other time, but there is no effect until its completion. Examples of this first type are preparing dinner, taking medicine, or fetching a laptop. The second type of activity affects the agent during its performance. There may be an expected duration, but when interrupted, the agent still partially benefits from its effects, for example sleeping, watching television, playing games or working.

3.4 Simulation of the User in the Model

Scenarios generally do not describe a user's whole day in detail, on a minute by minute basis. There are several activities which are not planned but are expected to happen during a day, or spontaneous reactions may occur to unexpected situations. The model described in this paper aims to simulate believable and realistic behaviour for agents. This behavior does not just follow a strict schedule; it is also based on the needs of agents, and their reactions to the current situations. Therefore, even when no scenario is used, the agent would perform routine activities (sleeping, eating, drinking etc.). On the other hand, if the schedule is unrealistically full, then the agent should be able to skip planned activities to meet their daily needs, as any rational person would.

This design creates a new problem: there are two independent decision-making processes which want to control the behaviour of the single agent in an incompatible way. The task of connecting these two processes to work together is complicated. It requires a third method to decide which should be preferred at each moment or whether one of the processes always has priority. In the second case, the preferred process has to be able to decide when to transfer the responsibility for selecting an activity to the other process.

The model described in this paper utilises a fuzzy cognitive map (FCM) modified for autonomous agents [27] as the primary decision-making method for the agent (Person). This method simulates the needs of the user and selects appropriate activities. It also reads the schedule to determine whether any activity is currently planned or shortly expected. If there is no immediate need to cover vital functions, then the FCM passes selection of activity to the schedule. The FCM handles daily needs including hunger, thirst, fatigue, hygiene, bladder, and boredom.

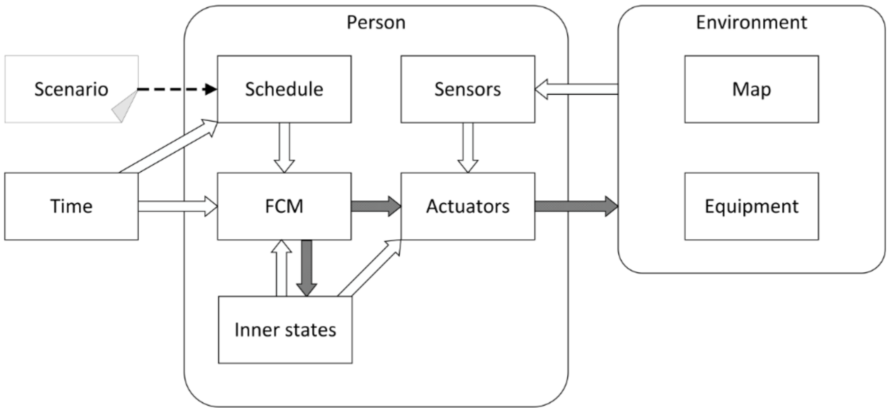

Fig. 4. Diagram of the Person agent

Fig. 4 shows the structure of the Person agent. The white arrows represent the flow of information, while the dark arrows express the direction of influence or control. The dashed arrow shows the importing of a scenario into the schedule. To make decisions, the FCM gathers data from the schedule and inner states. The schedule returns values representing whether there is an activity that should currently be in progress or is expected to start soon. Inner states provide values for all needs and the activity performed last time. The FCM is able to make higher-level decisions on what the agent should do. It selects a general group of activities and forwards it to the actuators, which select a specific action based on the current location or conditions. These lower-level decisions are made using simple rules which select a specific action and ensure the correct position of the agent. For example, when the agent is expected to eat, the algorithm checks the agent's corresponding level of need (hunger). If it is high, then the agent starts to cook a meal; otherwise, it eats a snack. If the agent is not in the kitchen, it starts moving to the relevant location.

4 Experimental Results

The experiments include an analysis of the behavior of the Person agent in the model under the scenario shown in **Table 6**. Since the user may begin days with different inner values, and activities vary slightly in duration, the final order and starting times of activities may differ each day, although the schedule is always the same.

Table 7 shows the course of a since day, in comparison with the scenario introduced in **Table 6**. Most of the activities were completed: some on time, some later. For example, in the afternoon, the agent had no time to continue reading the news because he/she had to make dinner. Later evening, the agent was tired and rested, ignoring a less critical activity in the schedule (playing games). However, he/she was able to accomplish this later.

Table 6. Scenario tested in the model

Start time	Activity	Probability
6:00	bathroom	0.95
6:15	medicine	1
6:30	exercise	0.7
6:45	medicine	1
6:50	shower	0.9
7:00	eat_breakfast	0.95
7:10	read_news	0.8
7:20	medicine	1
9:30	massage	0.9
10:30	swimming	0.9
12:30	eat_dinner	0.9
16:00	watch_tv	0.3
17:00	read_news	0.3
17:30	health_practice	0.8
18:00	eat_dinner	1
18:50	medicine	1
19:00	go_out_exercise	0.8

20:00	work	0.5
21:00	play_games	0.3
22:30	drink	0.8
22:45	bathroom	0.95
23:00	sleep	1

Table 7. Actual behaviour of the agent in a single day and comparison with the scenario

Time	Activity	Schedule comparison	Time	Activity	Schedule comparison
5:47	bathroom	13 min early	14:48	get_drink	not planned
5:58	medicine	17 min early	14:51	drink	not planned
6:03	take_snack	not planned	14:53	take_snack	not planned
6:05	eat_snack	not planned	14:55	eat_snack	not planned
6:11	watch_tv	not planned	15:01	watch_tv	59 min early
6:23	exercise	7 min early	16:27	get_drink	not planned
6:39	medicine	3 min early	16:30	drink	not planned
6:44	get_drink	not planned	16:32	watch_tv	resumed
6:47	drink	not planned	17:16	read_news	16 min late
6:49	watch_tv	not planned	17:17	health_practice	13 min early
7:00	eat_breakfast	not hungry (skipped)	17:41	read_news	resumed
7:00	read_news	10 min early	17:49	get_ingredients	required act
7:05	medicine	15 min early	17:55	make_dinner	required act
7:10	toilet	not planned	18:14	missed activity: read_news (3 minutes left)	
7:11	watch_tv	not planned	18:16	eat_dinner	16 min late
8:23	get_drink	not planned	18:32	get_drink	not planned
8:26	drink	not planned	18:35	drink	not planned
8:28	watch_tv	not planned	18:37	toilet	not planned
9:28	massage	2 min early	18:38	medicine	12 min early
10:29	swimming	1 min early	18:43	go_out_exercise	17 min early
11:29	get_drink	not planned	20:25	work	25 min late
11:33	drink	not planned	21:01	get_drink	not planned
11:35	get_ingredients	required act	21:04	drink	not planned
11:41	make_dinner	required act	21:06	rest	not planned
12:02	eat_dinner	30 min early	22:14	play_games	74 min late
12:18	toilet	not planned	22:29	get_drink	required act
12:19	watch_tv	not planned	22:32	drink	2 min late
13:09	get_drink	not planned	22:34	play_games	resumed

13:12	drink	not planned	22:42	bathroom	3 min early
13:14	wash_hands	not planned	22:53	play_games	resumed
13:40	toilet	not planned	22:58	sleep	2 min early
13:41	watch_tv	not planned			

Table 7 shows output data originating directly from the Person agent. However, the CHS does not have these outputs. A full exploration of the possibilities and limitations of such a system depends on its sensors, and there is no direct communication between the user and the system, even in the model. **Table 8** shows the information available to the system during the first few hours of the same day.

Table 8. Output data of the system

Time	Type	Location	Action
5:47:08	Move	Center	Person is captured by Center
5:47:13	Move	Hall	Person is captured by Hall
5:47:18	Move	Bathroom	Person is captured by Bathroom
5:48:03	On/off	Bathroom	Bathroom sink was turned on
5:58:03	On/off	Bathroom	Bathroom sink was turned off
6:03:03	Move	Hall	Person is captured by Hall
6:03:08	Move	Center	Person is captured by Center
6:03:13	Move	Kitchen	Person is captured by Kitchen
6:04:03	On/off	Kitchen	Fridge was turned on
6:05:03	On/off	Kitchen	Fridge was turned off
6:11:03	Move	Centre	Person is captured by Centre
6:11:13	Move	Living room	Person is captured by Living room
6:12:03	On/off	Living room	TV was turned on
6:39:08	Move	Centre	Person is captured by Centre
6:39:08	On/off	Living room	Person is in a different room than the TV, which was turned off by CHS
6:39:13	Move	Hall	Person is captured by Hall
6:39:18	Move	Bathroom	Person is captured by Bathroom
6:44:03	Move	Hall	Person is captured by Hall
6:44:08	Move	Centre	Person is captured by Centre
6:44:13	Move	Kitchen	Person is captured by Kitchen
6:49:03	Move	Centre	Person is captured by Centre
6:49:13	Move	Living room	Person is captured by Living room
6:50:03	On/off	Living room	TV was turned on
7:00:03	On/off	Living room	TV was turned off
7:05:08	Move	Centre	Person is captured by Centre
7:05:13	Move	Hall	Person is captured by Hall
7:05:18	Move	Bathroom	Person is captured by Bathroom
7:10:03	Move	Hall	Person is captured by Hall

7:10:13	Move	Toilet	Person is captured by Toilet
7:11:08	Move	Hall	Person is captured by Hall
7:11:13	Move	Centre	Person is captured by Centre
7:11:18	Move	Living room	Person is captured by Living room
7:12:03	On/off	Living room	TV was turned on
8:23:03	On/off	Living room	TV was turned off

The sensor log is much more detailed than the record of the person's activities. A whole day would cover several pages, and hence only the beginning of this day is presented in **Table 7**. Since both tables describe the same day, it is easy to compare what the CHS knows about person's activities and what the person actually did. For example, at 6:11, the Person agent decided to watch TV, and at 6:12 TV was turned on. At 6:23, the person stopped watching TV and started exercising. Since both activities took place in the same room and the person did not turn off the TV, the system did not notice a change until 6:39, when the person left the room (with the TV still on). The system then turned off the TV because no one was watching it.

5 Conclusion and Final Remarks

This paper describes an approach that uses scenarios to model interactions based on the routine procedures of inhabitants (users) of intelligent environments. In order to describe the various features of this use of scenarios in a comprehensible and illustrative way, a specific case study of a user (client) of a spa resort is introduced and provided with a technical background description. The model presented here of an intelligent environment was created based on the real-world spa resort of Trencianske Teplice (situated in the western Slovakian Republic) and covers both interior and exterior perspectives within a limited geographical area. Attention was focused mainly on the creation of various functionalities tailored to the user's needs, according to his/her daily routines and expectations of the services available in the spa resort. The scenario provided here shows a positive potential for improving the analytical and design phases of developing an intelligent environment, and the simulation model illustrates clearly how the scenario-based approach can be used to model the interactions between a user and an intelligent environment.

The main aim of this paper was to study the behavioural patterns of a user during his/her stay in an intelligent environment, illustrated here using the case of a spa resort. Particular attention was paid to which facilities were used and which functionalities were activated by the client. The client's activities and his/her interactions with the environment were monitored and stored in the activity log, allowing subsequent analysis and enabling the design of the intelligent environment to be tailored specifically to its users' needs. This approach allows this type of design to be completed faster, and environment to be equipped with functionalities which are specifically required and utilised by its target users. The final solution may also be cheaper, since it allows the elimination of unimportant or unused elements.

Acknowledgements. Financial support from GAČR (scientific project GA15-11724S DEPIES – "Decision Processes in Intelligent Environments"), and support from the Faculty of Informatics of the University of Hradec Kralove (specific research project "Socio-economic models and autonomous systems") is gratefully acknowledged.

References

1. Hodgson, G.M.: The ubiquity of habits and rules. Cambridge Journal of Economics 21, 663-684 (1997)
2. Banovic, N., Buzali, T., Chevalier, F., Mankoff, J., Dey, A.K.: Modeling and understanding human routine behavior. Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems, pp. 248-260. ACM (2016)
3. Kamara-Esteban, O., Azkune, G., Pijoan, A., Borges, C.E., Alonso-Vicario, A., Lopez-de-Ipina, D.: MASSHA: An agent-based approach for human activity simulation in intelligent environments. Pervasive and Mobile Computing 40, 279-300 (2017)
4. Kormányos, B., Pataki, B.: Multilevel simulation of daily activities: Why and how? 2013 IEEE International Conference on Computational Intelligence and Virtual Environments for Measurement Systems and Applications (CIVEMSA), pp. 1-6 (2013)
5. Helal, A., Cho, K., Lee, W., Sung, Y., Lee, J.W., Kim, E.: 3D Modeling and Simulation of Human Activities in Smart Spaces. In: Apduhan, B.O., Hsu, C.H., Dohi, T., Ishida, K., Yang, L.T., Ma, J. (eds.) 2012 9th International Conference on Ubiquitous Intelligence & Computing and 9th International Conference on Autonomic & Trusted Computing, pp. 112-119. IEEE Computer Soc, Los Alamitos (2012)
6. Mikulecky, P., Cimler, R.: Simulating Decision Processes in Intelligent Environments. In: Novais, P., Konomi, S. (eds.) Intelligent Environments 2016, 21, pp. 24-32. IOS Press, Amsterdam (2016)
7. Armac, I., Retkowitz, D.: Simulation of smart environments. IEEE, New York (2007)
8. Azkune, G., Almeida, A., Lopez-de-Ipina, D., Chen, L.M.: Combining Users' Activity Survey and Simulators to Evaluate Human Activity Recognition Systems. Sensors 15, 8192-8213 (2015)
9. Bonino, D., Corno, F.: Modeling, simulation and emulation of intelligent domotic environments. Autom. Constr. 20, 967-981 (2011)
10. Murakami, Y., Ishida, T., Kawasoe, T., Hishiyama, R.: Scenario description for multi-agent simulation. Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems, pp. 369-376. ACM, Melbourne, Australia (2003)
11. Grabisch, M.: Temporal scenario modelling and recognition based on possibilistic logic. Artificial Intelligence 148, 261-289 (2003)

12. Uchitel, S., Kramer, J., Magee, J.: Incremental elaboration of scenario-based specifications and behavior models using implied scenarios. ACM Trans. Softw. Eng. Methodol. 13, 37-85 (2004)
13. Pinheiro, F.A.: Preliminary thoughts on using situation theory for scenario modelling. V Workshop Iberoamericano de Ingeniería de Requisitos y Ambientes de Software, pp. 272-280 (2002)
14. Nikolai, C., Madey, G.: Tools of the Trade: A Survey of Various Agent Based Modeling Platforms. Journal of Artificial Societies and Social Simulation 12, 37 (2009)
15. Cavrak, I., Stranjak, A., Zagar, M.: SDLMAS: A Scenario Modeling Framework for Multi-Agent Systems. J. Univers. Comput. Sci. 15, 898-925 (2009)
16. Bauer, B., Muller, J.P., Odell, J.: Agent UML: A formalism for specifying multiagent software systems. Int. J. Softw. Eng. Knowl. Eng. 11, 207-230 (2001)
17. Jensen, K., Rozenberg, G.: High-level Petri nets: theory and application. Springer Science & Business Media (2012)
18. Paurobally, S., Cunningham, J., Jennings, N.R.: Developing agent interaction protocols using graphical and logical methodologies. In: Dastani, M., Dix, J., Seghrouchni, A.E.F. (eds.) Programming Multi-Agent Systems, 3067, pp. 149-168. Springer-Verlag Berlin, Berlin (2003)
19. Huget, M.P.: Agent UML notation for multiagent system design. IEEE Internet Comput. 8, 63-71 (2004)
20. Cicirelli, F., Furfaro, A., Nigro, L., Pupo, F.: Agent Methodological Layers in Repast Simphony. European Council Modelling & Simulation, Nottingham (2013)
21. Luke, S., Cioffi-Revilla, C., Panait, L., Sullivan, K., Balan, G.: MASON: A multiagent simulation environment. Simul.-Trans. Soc. Model. Simul. Int. 81, 517-527 (2005)
22. Kamiński, B., Szufel, P.: Verification of Models in Agent Based Computational Economics — Lessons from Software Engineering. pp. 185-199. Springer Berlin Heidelberg, Berlin, Heidelberg (2013)
23. Ormerod, P., Rosewell, B.: Validation and Verification of Agent-Based Models in the Social Sciences. pp. 130-140. Springer Berlin Heidelberg, Berlin, Heidelberg (2009)
24. Synnott, J., Nugent, C., Jeffers, P.: Simulation of Smart Home Activity Datasets. Sensors 15, 14162-14179 (2015)
25. Helal, S., Lee, J.W., Hossain, S., Kim, E., Hagras, H., Cook, D.: Persim - Simulator for Human Activities in Pervasive Spaces. 2011 Seventh International Conference on Intelligent Environments, pp. 192-199 (2011)
26. Buchmayr, M., Kurschl, W., Küng, J.: A Simulator for Generating and Visualizing Sensor Data for Ambient Intelligence Environments. Procedia Computer Science 5, 90-97 (2011)
27. Nachazel, T.: Optimization of Decision-Making in Artificial Life Model Based on Fuzzy Cognitive Maps. In: Weber, M., Minker, W., Mikulecky, P., Cech, P., Hunter, G., Egerton, S. (eds.) 2015 International Conference on Intelligent Environments IE 2015, pp. 136-139. IOS Press, Prague (2015)