# Using Training with Older People for Active Ageing in Time and Space *

Amedeo Cesta, Gabriella Cortellessa, and Riccardo De Benedictis

CNR - Italian National Research Council, ISTC, Rome, Italy,
{name.surname}@istc.cnr.it

**Abstract.** This paper introduces aspects of the authors' work within the "Città Educante" project that aims at exploiting advanced ICT technology to rethink the concept of learning environment and reshape it within a smart city perspective. In particular, the paper introduces the *LECTurE* module that, by relying on the timeline-based approach to automated planning, proposes contextualized lessons, as well as interaction requests, to its users. The lesson's content, specifically tailored for the older adults, is personalized taking into account users' psychophysiological aspects as well as geo-localization information and temporal constraints. After a generic introduction to the "Città Educante" project, the *LECTurE* module is presented and instantiated in two relevant use cases: the *on-site training*, in which the system is used as a support to the classical teaching methodologies within a classroom, and the *distributed training*, in which the technology aims to move and animate the teaching experience also "outside the classroom" with additional stimuli for the users during a practical experience (e.g., a real visit in a museum to complement a theoretical art history lesson). The paper, then, describes the implementation choices made for the realization of a first prototype, the reasons for choosing the timeline-based approach to planning as the underlying reasoning technology and the overall module architecture.

**Keywords:** active aging, learning environments, timeline-based planning, personalized interaction

## 1 Introduction

The "Città Educante" project[1] (the name means "city that educates" in Italian) aims at radically rethinking the learning environment through the application of the most advanced computer technology. The project proposes new educational approaches, enriching and innovating methods and tools, overcoming classical systems and the traditional "lessons" by providing learning in time (e.g., lifelong) and space (e.g., at school, in an outdoor environment, during leisure, etc.).

* Authors work is partially funded by MIUR under Cluster Program 2012. Send correspondence to *amedeo.cesta@istc.cnr.it*.

[1] http://www.cittaeducante.it

In this context, the theme of *learning* is framed in relation to the response to social challenges linked to the renewal of the educational system, to be achieved through the implementation of new learning/teaching models and/or the optimization of the existing ones on the various areas of life and knowledge, as well as new systems/evaluation processes, where technologies (platforms and web) become an enabling factor.

The authors' goal in the project has been the one of thinking an incarnation of the "Città Educante" for the continuous education of older people. In particular, considering the more recent experiences in the interaction of elderlies with complex machines [5], keeping in mind previous experience in training, specifically tailored for crisis managers [2], we are currently pursuing the goal of building a learning environment, called *LECTurE* (for Active Learning Environment for "Città Educante"), aimed at increasing the active aging and participation in social life of elderlies at home, in the community, and at work.

This paper represents an introduction to authors' effort in designing and building the *LECTurE* learning subsystem. After introducing the main ideas underlying the learning subsystem in Section 2, Section 3 introduces some technical background to illustrate the timeline-based technology upon which the system is based. Sections 4 and 5 describe a proposal for the modeling of students and lessons. A preliminary system architecture is proposed in Section 6. Finally, some conclusions and a discussion about future works close the paper.

## 2 Using AI to personalize lessons for older users

Inspired by the objective of reformulating the learning environment through the creation of platforms, services and ICT applications, we got in touch with several volunteering organizations addressing, specifically, elderlies' needs. Among the different organizations, in particular, Televita[2] is a volunteering association whose main objective is to maintain the elderlies active and motivated, leveraging upon individual aptitudes and/or competencies [4]. Televita's volunteers, in fact, are, themselves, elderlies who want to keep active by offering their abilities and competences to the organization. Although Televita's main activity consists in providing tele-assistance services (tele-friendship) and a 24h active helpline devoted to lonely elderlies who need support, it also manages several laboratories that involve elderlies both as attendees and "teachers". Examples include a computer lab, a tailoring lab, a cooking lab, an Italian language teaching for foreign people, etc. Furthermore, the association organizes cultural events as concerts, museum visiting, theatre, etc.

Among the offered services, we focused on two specific activities that were both in line with the "Città Educante" concept and suitable to be enhanced by Artificial Intelligence techniques:

- The *AttivaMente laboratory*: aims at keeping elderlies mentally active, so as to limit the cognitive decline associated with the advancement of age, by

[2] http://www.televita.org

proposing them cognitive stimuli. Such stimuli, mostly consisting in general culture quiz and/or crosswords, are proposed to the elderlies in a context similar to a school lesson. Specifically, by relying on some previous knowledge of the involved persons, as well as on their interactions during the lesson, a teacher, a volunteer himself, yet with more experience than others, controls the course's progress and slightly adapt it to the specific context's needs. Since the stimuli are predetermined, however, such adaptations tend to be limited to the possibilities of the case. The use of artificial intelligence techniques, in this case, could support the personalised delivery of stimuli by taking into account previous knowledge of the participants as well as their interactions with the AI system yet increasing, compared to the classical case, the personalisation capabilities.

– The *Art History lessons and cultural events*: analogously to the AttivaMente case, before attending to cultural events, some of the participants, according to their abilities and competencies, are asked to find information about some specific aspects of the event (e.g., a particular work of art within a museum, relevant historical happenings related to a visited site, etc.). Such information are then shared, in a "sort of lesson" with other participants in a classroom context and also outside during the event, enriching the overall knowledge of the group while encouraging the interaction among the members. Similarly to the AttivaMente case, such information may result to be limited and/or not customised to the specific members of the event. AI techniques, in this case, might offer the opportunity to enrich the cultural events experience by providing further personalised stimuli to the event participants, as well as interaction requests to test the level of engagement and actively stimulate them.

Combining both activities represents an opportunity for the elderly to keep themselves active while learning in time and space, and the use of AI can support the development of more effective and engaging learning experience.

### 2.1 The *LECTurE* learning subsystem

The idea of developing the *LECTurE* system was then born as a consequence of this field experience. Specifically, taking inspiration from a previous work [3], in which students were trained for managing crisis, the approach used within *LECTurE* is based on the idea of dynamically composing lessons through the use of a technology related to automated planning. In particular, starting from a static representation containing an high-level lesson track, initially stored in the database, a lesson is planned and dynamically adapted and personalized for the involved users. The idea of using the technology related to automated planning comes from the need to create a sufficiently extensive didactic experience to reproduce a large number of different situations which are, at the same time, characterized by a high variability of stimuli, aimed at increasing the involvement level of users. Automated planning, indeed, favors the generation of different lessons that would be too complicated to obtain with a simple pre-compilation of

stories. The timeline-based approach to planning [9], in particular, represents the unifying element of the various modules by ensuring the dynamic adaptability of plans by promoting experiential learning.

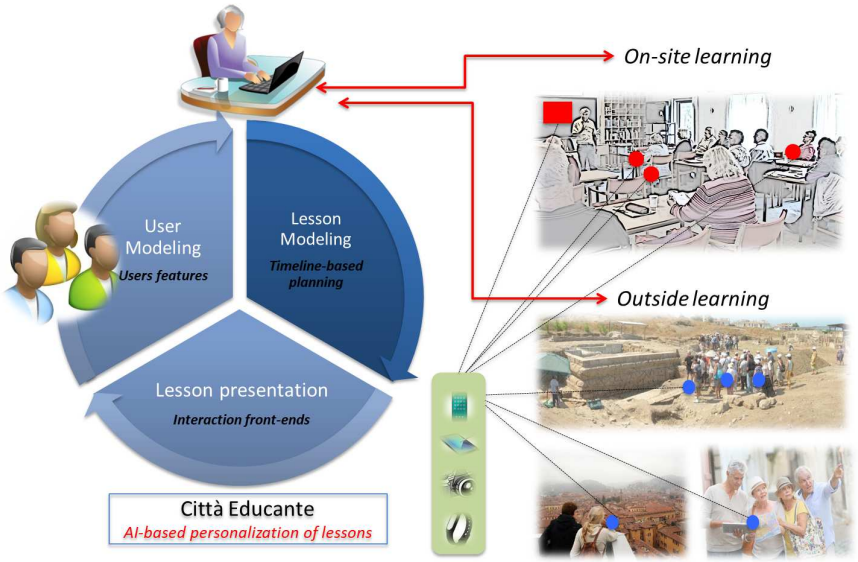


**Fig. 1.** The *LECTurE* general idea.

From a high-level point of view, the main modules of the system are described in Figure 1. In particular, it is possible to distinguish between two kinds of **involved users**: the *students*, i.e. a group of people, potentially, of any age, interested in using the learning services offered by the *LECTurE*, and the *teachers*, i.e. users with special privileges who have the opportunity to observe students, monitoring the progress of the lessons and of the overall learning environment. The above users interact with the *LECTurE* system which is composed of three functional blocks, intended as architectural subsystems, implementing the corresponding high-level functionalities: (i) the *user modeling*, whose goal is to create and maintain a user model and provide guidance for improving the learning process; (ii) the *lesson modeling*, whose role is to combine the information from the previous subsystem and to create the customized lesson as well as to control its evolution; (iii) the *lesson presentation*, whose purpose is to effectively represent the lesson.

It is worth highlighting that the proposed system provides users, whether students or teachers, the opportunity to change the learning environment's evolution in real time by interpreting their decisions. In fact, the architecture is based on a sense-plan-act paradigm implementing, in a continuous loop, the three primitives (a) *sense*, in which information is collected from sensors, (b)

*plan* where a world model is created using the information available to plan the next move to do, and (c) *act*, in which the move, chosen by the planning process, is actually executed. Furthermore, by mimicking the different cases implemented by the Televita organization, the *LECTurE* system provides training through different modalities depending on the chosen use case. In particular, it is possible to distinguish between two different modalities: (a) *on-site learning*, closer to the classical teaching, in which technology is used as a support to the teaching in a classroom, with the aim to create richer lessons, and (b) *distributed learning*, in which technology aims to support lessons outside the classroom during a practical experience. The following sections describe, more in detail, the above modalities.

### 2.2 On-site training

In the on-site training modality, the system can be used by a group of students, at the same time, previously contacted by the teacher. This mode represents an extension to the classical learning method in which a teacher teaches to a group of students. In this case, however, compared to the classical approach, the teaching is enhanced by the introduction, within the lesson, of the *LECTurE* technology.

Specifically, each lesson is instantiated by the teacher by defining the specific learning objectives. The system processes the lesson and presents the information to the students through the available tools. Students interact directly with the system, providing their answers to certain circumstances proposed by the system, and transmitting data from sensors available on the adopted devices (e.g. physiological parameters) enriching the users' models.

Depending on the sensors' inputs and on the students responses, both the teacher and the system can autonomously decide, based on the observations, whether and how to modify the current lesson. Finally, at the end of the lesson, the system can generate several reports, for example, one for each student, which can be used for debriefing purposes.

### 2.3 Distributed training

In the distributed training case, the system exploits different types of web technologies. This means that the lesson does not happen in a single physical room and is distributed among the students who are remotely connected to the system. The lesson can still be instantiated by the teacher defining the specific learning objectives but may have variable, potentially infinite, duration.

Students interact directly with the system, providing their answers to certain circumstances proposed by the system as well as transmitting data from the sensors available on the chosen devices (e.g., geographic location, physiological parameters, etc.). Sensor data enriches the users' models which, in turn, adapt the lesson to the students resulting in a highly personalized learning experience.

Again, based on the inputs from the sensors and on the students' feedback, both the teacher and the system can autonomously decide, based on the observations, whether and how to modify the lesson in progress. Similarly to the on-site case, the *LECTurE* system can generate, during the lesson, several reports, for example, one for each student, which can be used both for debriefing purposes and for a further tuning of the lesson.

## 3 Timeline-based planning to support dynamic lessons

Since most of the components of the *LECTurE* system strongly depend on temporal aspects, we have chosen to rely on a specific automated planning technique, called timeline-based, which allows to explicitly reason on time. Timeline-based planning, indeed, allows to reason about events in time and, hence, represents a valid tool for meeting our pedagogical needs. Planning a lesson, in particular, requires dispatching information at proper time. Additionally, reacting to users' interactions requires plan adaptation capabilities which can more hardly be achieved through other automatic planning techniques. Furthermore, the dynamic adaptation of the user profiles, which can take place on the different features that represent the user's model, can also be achieved through timelines.

As already said, timeline-based planning constitutes a technology that easily allows us to solve our problems. In order to better contextualize the choices that we have made and to better explain the different components of the system, however, it is worth introducing some basic formalism about constraint networks, on which timeline-based planning search strongly relies, and about the main concepts related to timeline-based planning.

The main ingredients of constraint networks are variables and constraints.

**Definition 1.** *A* variable *is an object that has a name and is able to take different values.*

A variable (whose name is) $x$ must be given a value from a set that is called the *domain* of $x$ and is denoted by $dom\,(x)$. The domain of a variable $x$ may evolve in time but is always included in a set called *initial domain*. Depending on the nature of these domains, variables can be distinguished between *continuous*, having an infinite initial domain usually defined in terms of real intervals, and *discrete*, whose initial domain contains a finite number of values.

**Definition 2.** *A* constraint *is a restriction on combinations of values that can be taken simultaneously by a set of variables.*

A constraint $c$ is defined over a set of variables which constitute the *scope* of $c$ and are denoted by $scp\,(c)$.

A structure composed of variables and constraints is called a *constraint network*.

**Definition 3.** *A* constraint network $\mathcal{N}$ *is composed of a finite set of variables, denoted by* $vars\,(\mathcal{N})$*, and a finite set of constraints, denoted by* $cons\,(\mathcal{N})$*, such that* $\forall c \in cons\,(\mathcal{N})\,, scp\,(c) \subseteq vars\,(\mathcal{N})$.

Finally, an *evaluation* of a constraint network is an assignment of values to some or all the variables. An evaluation is said to be *consistent* if it does not violate any constraint. An evaluation is said to be *complete* if it includes all the variables. Finally, given a constraint network, the problem of finding a consistent and complete evaluation is called Constraint Satisfaction Problem (CSP) (refer to [7, 8] for a comprehensive introduction to CSPs).

As regards timeline-based planning, the main data structure is the *timeline* which, in generic terms, is a function of time over a finite domain. Values on timelines are called *tokens* and are represented by temporally scoped predicates (i.e. predicates endowed with extra arguments belonging to the Time domain $\mathbb{T}$, either real or discrete). Two additional arguments indicate the timeline on which the token apply and the state of the token.

**Definition 4.** *A* token *is an expression of the form:*

$$n\left(x_0, \ldots, x_k\right) @ \left[s, e, \delta, \tau, \sigma\right]$$

*where $n$ is a predicate name, $x_0, \ldots, x_k$ are constants, numeric variables or object variables, $s$ and $e$ are temporal variables belonging to $\mathbb{T}$ such that $s \leq e$, $\delta$ is a numeric variable such that $\delta = e - s$, $\tau$ is an object variable representing the timeline on which the token apply and $\sigma \in \{Active, Unified\}$ is a variable representing the state of the token. A* token $n\left(x_0, \ldots, x_k\right) @ \left[s, e, \delta, \tau, \sigma\right]$ *asserts that $\forall t$ such that $s \leq t \leq e$, the relation $n\left(x_0, \ldots, x_k\right)$ holds at the time $t$ on the timeline $\tau$ if and only if $\sigma = Active$.*

The overall idea pursued in *LECTurE* consists in using such tokens for representing both the state of the users and the planned stimuli. For example, a relation $At\left(l\right)$, denoting the fact that a user is at a certain location $l$, can be enriched with temporal arguments $s, e \in \mathbb{T}$ and $\delta$ representing, respectively, its *starting time*, its *ending time* and its *duration*. The argument $\tau$ would represent the specific timeline responsible for modeling the user's position. The $At\left(l\right) @ \left[s, e, \delta, \tau, \sigma\right]$ token would now represent the user, whose position in time is described by the timeline $\tau$, being at location $l$ from time $s$ to time $e$ (for a duration $\delta$). It is worth to notice that in a grounded plan each token applies to a specific timeline, reflecting the intuition that tokens describe some aspect of the timeline (i.e. its state or behavior) in time. However, in general, the commitment to a specific timeline may not yet have been made and $\tau$ can be treated as any other variable of a constraint network. This applies, in particular, to stimuli, for which the recipient might be decided at planning time. More details about the state $\sigma$ of the tokens will be provided soon.

In order to reduce the allowed values for the tokens' constituting parameters, and thus decreasing the system's allowed behaviors, it is possible to impose constraints among them (and/or between the parameters and other possible variables). Such constraints include temporal constraints, usually expressed by means of interval relations [1], binding constraints between object variables as well as linear constraints among numerical variables (including temporal variables). Note that each token has an implicit temporal constraint that forces the

starting time to be non-negative (i.e. $s \geq 0$), an implicit temporal constraint that links its duration to its starting and ending variables (i.e. $\delta = e - s$) and an implicit temporal constraint that forces the starting time to be lower or equal than the ending time (i.e. $s \leq e$ that is equivalent to $\delta \geq 0$).

The set of tokens and constraints is used to describe the main data structure that will be used to represent nodes of the timeline-based search space: the *token network*.

**Definition 5.** *A* token network *is a tuple* $\pi = (T, C)$*, where:*

- $T = \{t_0, \ldots, t_k\}$ *is a set of tokens.*
- $C$ *is a set of constraints, required to be consistent, on the variables of the tokens in* $T$*.*

As regarding the $\sigma$ state variable associated to each tokens, its value has to be decided by the solving procedure. Specifically, each token either (i) *unifies* (i.e. the $Unified$ value is assigned to its state variable $\sigma$) with an already existing active (i.e. whose state variable $\sigma$ assumes the value *Active*) token assuming exactly the same value, i.e. the two tokens have the same predicate name and their arguments are pairwise constrained to be equal or (ii) is *activated* (i.e. the *Active* value is assigned to its state variable $\sigma$) and some "rule" is applied. This kind of rules is generalized in a concept usually called *compatibility*.

**Definition 6.** *A* compatibility *is a tuple* $c = (name\,(c)\,, R\,(c))$*, where:*

- $name\,(c)$ *is the* master *(or* reference *or, even,* trigger*) predicate and is an expression of the form* $n\,(x_0 \ldots x_k)$*, where* $n$ *is a unique predicate symbol with respect to a timeline (i.e. no two compatibilities in a given timeline can have the same predicate symbol), and* $x_0 \ldots x_k$ *are its associated variable symbols.*
- $R\,(c)$ *is a* requirement*, i.e. either a* slave *(or* target*) token, a constraint among tokens (possibly including the* $x_0 \ldots x_k$ *variables), a conjunction of requirements, a disjunction of requirements or a preferred requirement.*

Compatibilities define causal relations that must be complied to in order for a given token to be valid. Whenever a token, having a predicate name $n$, is activated, the master $name\,(c)$ of a compatibility $c$, having the same predicate name $n$, is substituted by the token and the requirement $R\,(c)$ is added to the token network. The idea underlying the unification process is equivalent to saying, in a sense, that the unifying token has already been demonstrated to be valid. Finally, it is worth to underscore that the compatibilities may often involve tokens applied on different timelines, thus allowing to synchronize concurrent activities on different domain components.

The last aspect to be addressed is the definition of the timeline-based planning problem which can rely on the above requirement concept.

**Definition 7.** *A timeline-based* planning problem *is a triple* $\mathcal{P} = (\mathcal{T}, \mathcal{C}, R)$*, where:*

– $\mathcal{T}$ *is a set of timelines.*
– $\mathcal{C}$ *is a set of compatibilities.*
– $R$ *is a requirement, i.e. either a token, a constraint among token arguments, a conjunction of requirements, a disjunction of requirements or a preferred requirement.*

## 4 Modeling the Students

By pursuing the overall goal of enhancing the learning experience, it is necessary to keep a user model up-to-date in order to consider how their emotional, psychological, physiological and geographical parameters can influence the learning process. Specifically, the student modeling has three main objectives:

– Select, model and monitor relevant human factors, as well as psychological, physiological, or other user-related variables;
– Develop a model that can represent the user's profile;
– Provide a high level guidance for customizing learning objectives.

In particular, the user model must consider psycho-physiological aspects (such as heart rate, personality traits, etc.), user-related aspects (e.g. geo-localization) and pedagogical parameters such as the different learning modalities. Its role consists in feeding the lesson model allowing the customization of the content of the lesson. Specifically, user profiles are stored on timelines in order to be retrieved later, interpreted or updated during the lesson. Depending on the particular status of the user, by relying on compatibilities, this module establishes custom learning paths with different difficulty levels tailored on the specific users. Each user has associated a set of timelines for dynamically maintaining this information in time. Figure 2, for example, shows the modeling of the user Alice by means of two timelines representing, respectively, her location in time and her performance. Tokens on the location timeline, as an example, have the form $At\,(x, y)\,@\,[s, e, \delta, \tau, \sigma]$ and represent the user $\tau$ being at coordinates $\langle x, y\rangle$ from time $s$ to time $e$. At regular intervals, the GPS position of the user is found and, in case of significant variations, a new token is added to the timeline so as to keep updated the position of the user in time.

Similarly, other timelines are used to keep updated the information about other features, not necessarily extrapolated from sensors. The set of considered relevant factors can be related, for example, to personality traits, past experience, the perceived effectiveness in performing complex tasks, current assessment, perceived stress, or anxiety. The use of Blue-tooth bracelets or bands for heart rate measurement may allow the extraction of physiological values such as heart rate or heart variability. Finally, it is possible to leverage on geo-localization services. The initial evaluation of these variables, used as a baseline to initialize the didactic experience and as a reference point for subsequent measurements, can be done through the use of standardized psychological questionnaires or physiological measurements performed off-line before the lesson. It is worth to notice, however, that the profile of a student can also be updated exploiting the

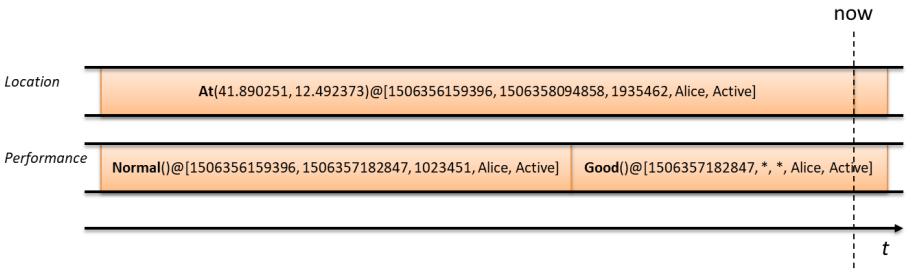


**Fig. 2.** Modeling the students through timelines.

interactions of the user with the system asking users, for example, to answer to sporadic questions.

Among the dynamic parameters, the student *performance* is monitored and observed by recording input data such as the different actions taken by the users. Specifically, this information is processed and interpreted in order to plan the subsequent actions of the lesson as well as to support the debriefing phase. By processing the users' profiles, the system generates a user model that is constantly updated to perceive and represent significant changes in the emotional state (note that parameters can generally change over time). In addition, students' performance is analyzed and processed in order to gather further usable information to further customize the lesson. The teacher can access this information in order to supervise and control the customization. For this purpose, this component can provide guidance on how to customize the lesson. Personalization of a training course can therefore be done automatically, but it can also be suggested to a teacher who independently decides whether to adapt the training course (i.e. according to a mixed-initiative style).

Since the different users are characterized by properties that are closely related to their role, their work or, more generally, their emotional state during the lessons, each user that interacts with the system feeds the *LECTurE* system with personal data. The system, in turn, builds a user model that is used to synthesize custom lessons responding to the specific status of each student. The output of this process is passed to a second module that, on the basis of these indications and on the particular didactic path chosen, synthesizes a sequence of appropriate stimuli for the group (shared information between the different students in a lesson) and for the individuals (tailored information for a specific student).

## 5 Modeling the Lessons

Lesson modeling is the key feature of the *LECTurE* system since it creates and manages the network of events (i.e. a token network as described in Section 3) that guides the entire learning session. The purpose of this feature is twofold:

on the one hand, the module maintains static information, a.k.a. *domain theory* (i.e. the set of compatibilities), needed to represent relevant system information like, for example, the procedures for managing the different responses to stimuli. On the other hand, this module is able to develop a model that describes the combined effect of static information, students' actions, teachers' adjustments, and information from the user model.

Within the *LECTurE*, planning goals are characterized by high-level events that represent the abstract model of the lesson. Specifically, the lesson is represented as an abstract plan that plays the lead role. The different events of a lesson are represented through tokens causally linked by means of compatibilities. In order to represent a lesson and to manage user interactions with the system and their associated consequences, a *Lesson* Timeline, containing the high-level events that may occur during the lesson, represented through tokens, is instantiated. Each user has associated an *Activity* Timeline representing their performed actions and allowing the system to compute the actions' consequences. Through the interaction with the domain theory, the *Lesson* timeline generates sub-goals, allowing to maintain a high-level vision for the teachers while providing sufficient granularity of the information sent to the students.

Specifically, the teacher loads the chosen lesson from the database and the planner, on its own behalf, works on the timeline representation to create a lesson (i.e. the set of events, positioned over time, communicated to students like videos, text messages, questions, etc.). Once the planner has reached a solution, given the learning objectives, the plan is executed by sending messages to the lesson presentation module according to the progressive order of the scheduled events. Some of these messages are sent to all the participants in the lessons or to the individual users to stimulate them or to ask for interaction through questions, or even to evaluate their emotional state so that they can feed the user's model.

Once the teacher has loaded a lesson and the planner has arranged the tokens over time, the plan is ready to be executed. Some of these tokens represent stimuli and requests for the students. In order to foster interaction and collaboration with other users, the distributed information may be partial, requiring students the need to send messages to other students so that they can build an overview and respond appropriately to the challenges posed by the system fostering cooperation.

The goal represented in Figure 3 through the token $Colosseum\,()\,[s, e, \delta, \tau, \sigma]$, for example, is used for representing an education unit related to the Colosseum. By applying its associated compatibility, the education unit is expanded, through subgoals, into several information units arranged over time to be delivered to the different users involved in the lesson. Such information units, instantiated through text, videos, etc., include notions related, e.g., to the building time of the Colosseum, to its microclimate, to its use as a source of materials for other buildings, etc. Additionally, among the subgoals, questions are intended to check the learning efficacy of the users so as to adapt, consequently, the lesson. In particular, answers are considered as "actions" performed by the users. Such actions are represented by tokens on the *Activity* timeline associated to each

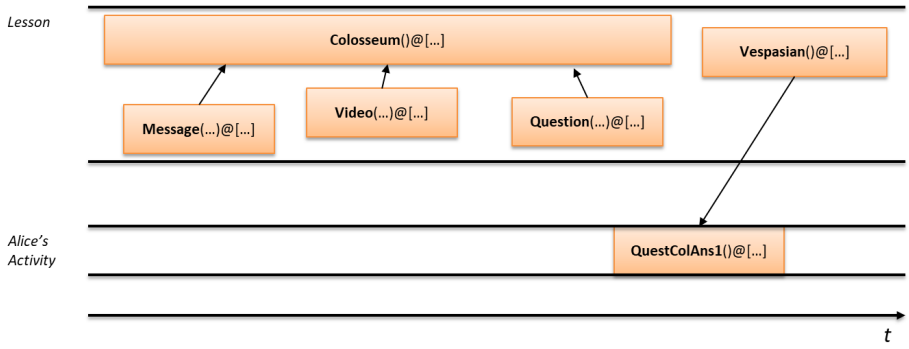


**Fig. 3.** Modeling the lessons and users' activities through timelines.

user and, as any token, require the application of a compatibility. It is through this mechanism that it is possible to change the system behavior by means of a plan adaptation. Timeline-based planning results to be particularly suitable for this kind of dynamic adaptation without having to re-plan from scratch.

As an example, a question related to the Colosseum might be the following:

> Under which dynasty was the Colosseum built?
>
> – Flavian
> – Constantinian
> – Julio-Claudian

Suppose, for example, that the user Alice answers (correctly) with the first option, the compatibility associated to the $QuestColAns1\,()$ predicate, corresponding to the first available answer for this question, is applied. Such a compatibility might be, for example:

$$QuestColAns1\,()\ \{v : lesson.Vespasian\,() \wedge v.start >= end + 300000\}$$

As a result, the $Vespasiano\,()$ education unit is added by the reasoner to the lesson and constrained to start at least 300000 milliseconds (i.e. 5 minutes) after the end of the answer. As any other token, the new token requires to be activated by means of its associated compatibility resulting in the introduction of further stimuli into the lesson as, for example, the following information about Vespasian:

Vespasian was Roman emperor from AD 69 to AD 79, the fourth, and last, in the Year of the Four Emperors. He founded the Flavian dynasty that ruled the Empire for 27 years.

These questions may have a maximum response time in terms of minutes, or days. In case of non-response, the system can choose a response to the user's seat, possibly decreasing the student's assessment. Alternatively, it is possible to set up a "hurry-up" mode that makes a countdown appear to increase the tension and the engagement of the students.

This form of reasoning is generalized in various contexts according to the topic of the lesson. Furthermore, the lesson modeling takes into account the customization aspects as suggested by the user modeling. An example, suppose a student is in the proximity of Vatican City, his/her profile is updated through geo-localization information and, through the application of a compatibility, synthesizes the following question:

Who painted the Sistine Chapel?

- Donatello
- Michelangelo
- Raffaello

Suppose that the student answers (correctly) with the second option, the dynamic adaptation of the plan, depending on the current lesson, could generate new stimuli such as, for example, the following information on the Sistine Chapel:

The Sistine Chapel, dedicated to Mary Assumed in Heaven, is one of the most famous cultural and artistic treasures of the Vatican City, inserted in the path of the Vatican Museums. It was built between 1475 and 1481, at the time of Pope Sisto IV della Rovere, from which he took the name.

Additionally, a positive reinforcement can be sent to the student increasing his/her score, resulting in new questions of increasing complexity.

Finally, as already briefly mentioned, it is worth to notice that since it is not possible to predict all the training courses during the lesson design phase, *LECTurE* provides the teachers the opportunity to modify the lessons in an incremental way, adapting the lesson to unpredictable behavior of the users. Alternatively, the teacher can manipulate the execution of the lesson so as to force specific predefined paths.

## 6 The System Architecture

The *LECTurE* system software architecture is described in Figure 4.

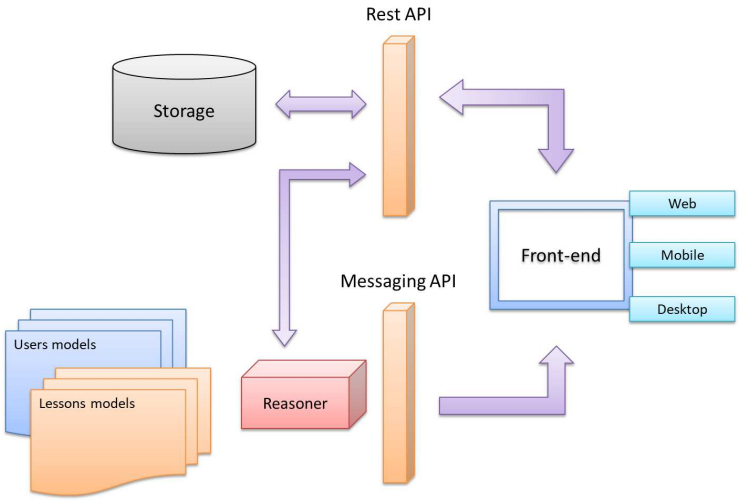


**Fig. 4.** The *LECTurE* main modules.

Specifically, the architecture contains a module for the permanent storing of relevant information. A reasoner, constituted by a timeline-based planner [6], works to keep in mind the different models, specified by means of compatibilities, of the users and of the lessons. A front-end module deals with providing graphical interfaces (web, mobile, and/or desktop) to the users. The communication among the different modules happens by means of two different software interfaces:

- A REST interface allows the creation, updating, and viewing of permanently stored information. The same interface is used to notify any changes to a user's profile (e.g. the user has moved around a site of historical interest) or possible actions by the user (e.g. answering a multiple-choice question) or, furthermore, messages sent to other users.
- A messaging interface, made through WebSocket and/or MQTT, allows to send different types of messages to the users. Examples of messages may be historical information related to, for example, the current user location, multiple response questions, questionnaires, maps, questions to detect the psychological state of the user, etc.

## 7 Conclusions and Future Works

This paper introduces the *LECTurE* system as an AI-based learning environment specialized so as to support active ageing. The paper defines the *LECTurE*'s initial functionalities, the architectural entities and a specification of the technological components. We have introduced the general idea of supporting older people in maintaining themselves mentally and physically active being helped by some intelligent technology both during a class and during excursions. The ICT

intelligent core makes use of timeline-based planning technology to represent key ingredients, create a baseline "lesson" and dynamically adapt it to integrate both personalised stimuli and requests to the users over time. The constant contact with Televita's volunteers is allowing the transition from a lab prototype to an incrementally more robust version of the system to be tested in realistic scenarios.

## References


1. Allen, J.F.: Maintaining Knowledge about Temporal Intervals. Commun. ACM 26(11), 832–843 (1983)
2. Bacon, L., Cesta, A., Coraci, L., Cortellessa, G., De Benedictis, R., Grilli, S., Polutnik, J., Strickland, K.: Training crisis managers with PANDORA. In: Proceedings of the 20th European Conference on Artificial Intelligence. pp. 1001–1002. IOS Press (2012)
3. Cesta, A., Cortellessa, G., De Benedictis, R.: Training for Crisis Decision Making - An Approach Based on Plan Adaptation. Knowledge-Based Systems 58, 98–112 (2014)
4. Cesta, A., Cortellessa, G., De Benedictis, R., Fracasso, F.: A tool for managing elderly volunteering activities in small organizations. In: Esposito, F., Basili, R., Ferilli, S., Lisi, F.A. (eds.) AI*IA 2017: Advances in Artificial Intelligence (2017)
5. Cortellessa, G., Fracasso, F., Sorrentino, A., Orlandini, A., Bernardi, G., Coraci, L., De Benedictis, R., Cesta, A.: ROBIN, a Telepresence Robot to Support Older Users Monitoring and Social Inclusion: Development and Evaluation. Telemedicine and e-Health (2017)
6. De Benedictis, R., Cesta, A.: Integrating Logic and Constraint Reasoning in a Timeline-based Planner. In: AI*IA 2015 - XIVth International Conference of the Italian Association for Artificial Intelligence (2015)
7. Dechter, R.: Constraint Processing. Elsevier Morgan Kaufmann (2003)
8. Lecoutre, C.: Constraint Networks: Techniques and Algorithms. Wiley-IEEE Press (2009)
9. Muscettola, N.: HSTS: Integrating Planning and Scheduling. In: Zweben, M. and Fox, M.S. (ed.) Intelligent Scheduling. Morgan Kauffmann (1994)