EAI Endorsed Transactions
on Creative Technologies
Research Article EAI.EU

CITE – Content Interaction Time and spacE: a hybrid approach to model man-robot interaction for deployment in museums

Armelle Prigent[1,*], Arnaud Revel[1]

[1]L3I Lab - University of La Rochelle

Abstract

In this paper, we present a generic model, CITE (Content-Interaction-Time-spacE) devoted to the development of interactive applications for places of culture such as museums or theaters. The idea is to introduce new kinds of interactions with visitors while preserving the quality of the information they can acquire into the museum. We firs present two previous works involving respectively a bottom-up model (LINK H/R) in the context of science-art interaction, and a top-down model (PLUG) which was an interactive game in a museum. The CITE model is then presented as a trade-o between those approaches: details are given concerning the Content and Interaction management; and an extension is considered concerning Time and spacE handling.

Received on 01 September 2016; accepted on 03 September 2017; published on 09 October 2017
Keywords: class file, LaTeX 2$_\varepsilon$, *EAI Endorsed Transactions*

doi:10.4108/eai.8-11-2017.153336

1. Introduction

Computer science and robotics have long been kept apart from cultural issues since they were considered as prototypes of dehumaniza tion. Paradoxicall y, the represen tation of computers and robots have alw ays been very presen t in liter ature and movies. It is well known, for instance, that the term robot itself has been introduced in a novel by Karel Čapek during the 20's. And there is no need to men tion HAL 9000 the mad computer in Kubrik's "2001, a space odyssey" movie or R2D2 and C3PO, the star robots of "star wars".

Step by step, computer science has emerg ed in the cultural fie d and has became a new tool to facilita te the crea tion of works of art. Actuall y, in music, computers are nowada ys as common instrumen ts as violins were in the 17th cen tury. More recen tly, robots has begun to be introduced in choreogr aph y such as in the "Robot !" show choreogr aphed by Bianca Li.

But cul ture does not only mean art: it has a wider sense that implies a larg er an heterog eneous public. Museums, for instance, are trying to give an access to cul ture to a larg e audience but they also convey the will to transmit knowledg e.

In this article, we are interested in considering what kind of robotic and computer models shoul d be dev eloped to allow the introd uction of robots in museums, temples of cul ture, introd ucing new inter actions with visitors, especiall y young ones, without degr ading the quality of knowledg e they can acquire.

Traditionall y, two kinds of approaches are proposed for elabor ating conception models: bottom-up and top-down. In the firs part of this article we presen t two previous experimen tation in rela tionship with arts and cul ture: the firs – Link H/R adopts a bottom-up approach, the second – PLUG, top-down consider ations. Both has found difficul ties to fulfil their goal en tirel y. In the second part of this article, we propose the CITE model, adopting a mixed approach, as a trade-o ff betw een bottom-up and top-down approaches.

2. Previous works

*Corresponding author. Email: armelle.prigent@univ-lr.fr

2.1. Link H/R: a bottom up approach

Context. Link Human/Robot is the result of an interdisciplinary collaboration between an artist, a dancer, a musician, a drawer and a roboticist. This project crosses different angles and expertise from those different fields. It is interdisciplinary by essence. The dialogue between the different disciplines develops three dimensions: social, scientific and artistic. The premises of the project was the common interest of the artist and the roboticist for the work of Varela [21] and the concept of autopoiesis.

A system is "autopoietic" if it can reproduce and maintain itself. Varela claims that autopoiesis, autonomy and cognition are intrinsically linked with each other. Originally developed by Maturana and Varela in the field of biology, the concept has been enlarged to artificial intelligence (especially what is called "nouvelle AI" – see for example [1, 4, 18]) and embedded robotics (for instance [17]).

The project by itself is a choreographic performance where a dancer and the humanoid robot Nao developed by Aldebaran Robotics are facing each other. Based on a turn taking imitation process inspired from the seminal work of J.Nadel [11], each agent builds its own choreographic motor repertoire through an accumulation of gestures. In that sense the robot (respectively the dancer) follows an accelerated developmental process equivalent to the psychological development of children [16].

Initially, we intended to develop a process where shared choreographic phrases and kinds of mutual interpretations and reciprocal improvisations would emerge. The robot, respectively the dancer, would not mime each other: they would react, interpret, and they would take the choreographic material from each other. This would have been a kind of dialogue with mutual influence.

In practice, the constraints imposed by the necessity to present the choreographic performance in front of a public has led to "guide" the development of the robot and to artificially sequence the different steps of the choreography. Therefore, the original robotic control architecture, that we would have liked to follow Varela's autopoietic principles has turned into a series of individual behavioral components manually triggered by a human being in order to respect the timing imposed by the choreography.

Control architecture. Initially, we intended to use an adapted version of the PerAc (Perception- Action) control architecture [6] to pilot the robot. This biologically inspired architecture has been proposed by Gaussier and Zrehen as a generic neural building block for robotic controllers. In a bottom-up perspective, the PerAc architecture is composed of two data streams corresponding to perception and action flows.

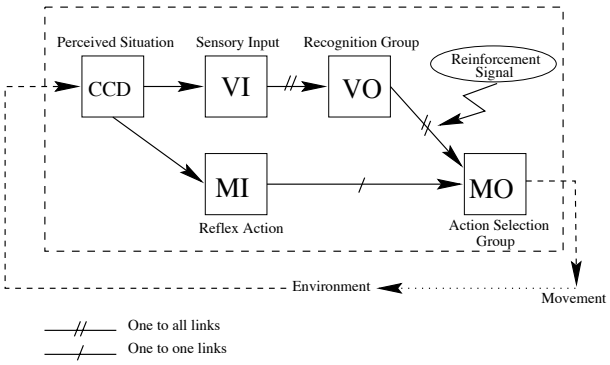

Figure 1. The PerAc architeture.

The lower layer is a kind of "reflex" mechanism directly controlling the actions thanks to the raw information extracted from the perception input. The upper layer uses a learning mechanism to perform recognition of processed elements extracted from the perception flow. Therefore, it allows to learn associations between the recognition of a specific perceptual element and a particular action and thus extend the reflexe behaviors. The main principle of the PerAc architecture, in a dynamic system perspective, is that the system evolves due to the dynamic interaction between the robot and its environment and can reach a behavioral equilibrium [3]. This control architecture has been used in many domains of robotics such as: indoor and outdoor navigation and planning [5, 7, 14] object recognition [10] and in particular in imitation [2, 15] which is the closest to our consideration.

Ideally, we would have liked to develop a single neural architecture allowing to learn by itself how to interact with the dancer and to learn from her a new action repertoire in order to progress and exchange via body communication. However, practically, this would have required much technical development time to integrate all the behaviors in the controller, but also, robotic developmental time for the robot to be able to incrementally learn (which is a long and uncertain process) the movements from the dancer.

For scenaristic and narrative reasons, we were then forced to chain the different behaviors composing the whole choreography. All the behaviors were thus conceived as small independent PerAc blocks with their own timing and autonomy. However, the orchestration of the blocks was delegated to a human supervisor triggering the behaviors when needed.

This is of course not adequate since the human is performing a major part of what the control architecture should have done: planning and scheduling the actions. The main drawback is that no overview of a current situation is available in order to trigger the appropriate behavior in a "top-down" approach.

2.2. A top down approach: PLUG

PLUG was an interdisciplinary project that intended studying technologies for ubiquitous games and their sociocultural, economic and industrial acceptability .

Context.. Two iterations were needed to develop this game. In the firs one, among the 4000 items of the museum 's (*musée des arts et métiers de Paris*) collection, a set of 16 items were selected to compose the content of the game. These objects were split into into four collections and virtual playing cards were associated with them. These virtual playing cards were stored in RFID terminals. The players used mobile phones to read, exchange or drop cards. The game objective was to gain as many points as possible by completing the collection (explore the terminals, exchange cards with other players, and answer quiz).

The second prototype of the project aimed at including more perv asiveness and ubiquity in the game. For that purpose, players had been provided with equipment as sensors to measure physiological data. These data were integrated into the game to change the course according to the bio-physical condition of the players. In comparison with the firs iteration, the game design was more directed towards the acquisition of scientifi contents on the objects than a simple card collection.

The goal of the game for player was to discover the historical and technical "memory" of the objects :

- Historical, societal memory: knowing to whom is due a particular invention? Why this innovation appeared at that time?
- Technical memory objects: what where the problems solved by this innovation?

At the end of the game session, players must have understood the technical aspects but also the reasons that led to the creation of this object, as its relevance to the aspirations of society that is contemporary to him.

This game is composed of a set of enigmas for players: a set of intermediate elements of discovery (ID) associated to a target object (TO). First, the research begins with a location enigma (the team has to fin the room in which is located the ID). When the team has validated its position on entering the room, another enigma is provided on the iphone and the team has to seek and fin the object in the room. Then the team validates the results of his research before looking for another ID.

Adaptive execution. The proposed adaptive engine is based on an architecture dedicated to performing real-time observation of the players, and making decisions on the scenario in in order to offer the best experience based on their observed behavior. This architecture is based on a model of this experience, including both the representation of the behavior of the players (through situations where it can be) and the decisions to be implemented to adapt the experience to their particular cases (through a set of rules for adaptation). Here, the challenge is to integrate the mechanism of observation-decision in the game server on which players connect.First, we have to insert in game engine, observation's points on the low-level events to monitor the behavior of the players (for example, we observe events like "enter a room", "open a new enigma. "..). Then, we store them in a database that is used by the observation module to build, in real- time,the state symbolic that represent each player. Interactivity

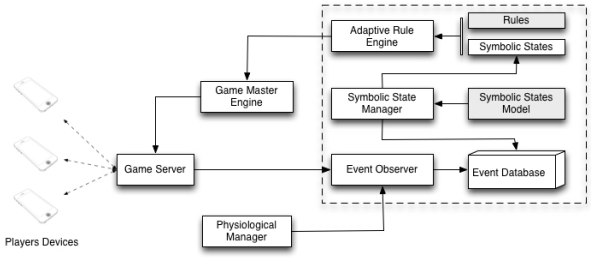

Figure 2. The Plug architecture.

should be considered as a feedback loop in which each involved entity adapts itself to the behavior of the other. Maintaining a player in a given narrative framework can be complex, particularly in the context of a pervasive multiplayer game. The following behaviors are to be supervised because they can endanger the experiment.

- Progression: slow, lack of motivation, misunderstood of gaming rules, blocking
- Social attitude: generous or competitive player (acceptance or refusal of systematic exchanges ...), independence (request for help, too much or too little), proximity with another team
- Strategy: imbalance in the strategies employed (the player promotes a way of playing compared to another).

Then, the adaptive engine performed observation and proposed adaptive events. A firs step is to receive events from the "client game logic" (player's behavior in the game) and the received sensor signals. These was transmitted to a server. This last proposed real- time modific tion of the game execution, according to data interpretation. A game master selected adaptations to be performed among these proposals to ensure game dynamic.

Symbolic states and interaction model. The proposed symbolic states are vectors of states built from parameters of the experiment. Thus, based on the

observations, the manager of symbolic statements builds a state $S_p =< PS, PL, St, So >$ for each player p with :

- PS: the speed of progression (blocked, slow, normal or fast)
- PL: the level of progression (low, normal or high)
- St: the level of stress (quiet, calm or excited)
- So: the level of social interactions (competitive, social or generous)

If it is necessary to know the state in which the player is, it is also important to analyze his sequences of actions. Indeed, in the PLUG project, we would have liked to be able to know the past actions of the player and his potential futures actions. The second advantage of such a model stands in the ability to authorize or inhibit an interaction in the light of the observed state.

In order to model the user's actions, we proposed a first approach that consist in representing all the possible actions (whether those permitted to the user or those executed by the system) through automata networks. This method is based on a reachability analysis performed through the model-checking tool UPPAAL to generate the optimal path to an expected state [12]. Although this method of adaptation has been proven effective, she becomes difficult to be used in the case of a system that handles a large number of states and interactions. Indeed, building a complete system can be tedious for the designer.

The major difficulty stands in the representation of all the possible interactions between the user and the narrative entities manipulated by the system. The model must guarantee a high level of modularity, reusability and needs to be scalable enough to meet potential changes. In addition, supervision models requires having a formal model from which it is possible to extract the most relevant paths.

3. Hybrid approach: the CITE architecture

In our innovative work, we propose a multi-dimensionnal approach: the CITE model (Content, Interaction, Time, spacE) will represent the different dimensions of the experience. The first one will represent all the contents handled in different formats (some text shown on screens, sounds, images, videos ...). The second dimension concerns the interaction represented as a protocol, based on the work conducted by [13] (timed automata networks).

The time taken into account, in the third dimension of the model, is multiple. First there is the time of the actors (it can be the visitor or different interaction elements), then the time for each narrative situation and finally the overall time of the experience. The last dimension concerns space and can model the various places of dissemination of content or execution of interactions.

This model has two advantages. It will be the basis of qualitative analysis and improvement of the scenario. It will also move towards a generic methodology of bringing into use of such an experience in different places of heritage.

In the rest of this section, we will first present the already implemented CI part of the model then give an overview our hybrid approach and finally give perspectives on how the model should be extended to take Time and spacE into consideration.

3.1. CI model: A situation-based approach for interactive storytelling

For the two dimensions C and I, we have proposed a high-level model for the representation of the player's states and possible actions in each state. Inspired by narratology, this situational model is implemented on three levels.

We proposed an approach based on an extension of I/O automata networks [9, 20]. The main idea of this model is to represent narrative situations that will be automatically computed from the behaviors of each entities acting in the situation. This model is inspired from narratology theory and uses a part of the vocabulary of [8]. The principle is based on the definitio of modular behaviors for the entities of the system, the automatic synthesis of the automaton representing entity behavior and finally the definitio of narrative situations involving these entities. The global system model is constructed by instantiation of the generic entities define in an *abstract layer* toward an *implementation layer* (Actant → Actor; Situation → Scene) and the definitio of scenes in which these entities operate. We also provide a supervision model, allowing the designer to specify the execution order of the previousl y define scenes, by applying a set of conditions. The last layer, here, represent the possible sequences of scenes exection through a directed graph of scene implementations called "plots".

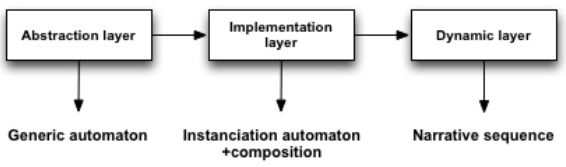

Figure 3. The three layers approach.

Abstraction layer.. In our three step approach, the abstraction layer is the one that help the designer to describe a set of narrative entities that will be involved

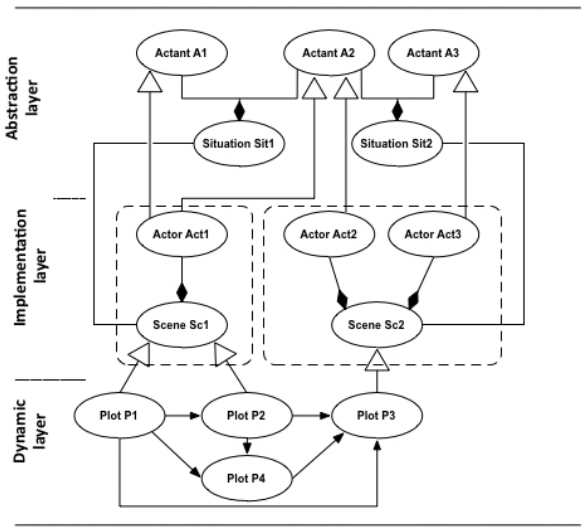

Figure 4. The situation based approach.

in situations. It defines the types roles (actants) and generic situations. We consider here the tuple (A, S, V_G) composed of A a set of actants, S a set of situations and V_G a set of global integer variables, that can be manipulated by all the actants. Let V_g^e be a valuation of all global variables in the abstraction layer (here called state vector).

Each actant is described through local variables and actions. An action is an abstract concept that is specialized as a reflexive action, an opposite action and an interaction. This extension allows representing particular patterns of automata. An action may involve the actant which it originates, or trigger a behavior in another actant: this is called interactions between actants. This offers the possibility of adding a synchronization character to another behavior.

Implementation layer.. Entities defined in the abstraction layer are generic. This second layer, called **implementation layer**, is used to implement generic entities defined in the abstraction layer into concrete entities. Thus, actants are instantiated by *actors* and situations by *scenes*. This mechanics of instantiation allows to create several actors from a single actant model and allows the multiple inheritance (an actor implements the behavior of several actants). We define the instantiation layer by the following tuple (A_{cr}, S_c) composed of a set of actors and a set of scenes. **Actor** can implement the role of one or more actant(s). Thanks to inheritance, the actor possesses behaviors it then implements. Multiple inheritance allows the designer to specialize an actor according to the behaviors of different actants it implements.

Dynamic layer.. This layer defines the execution scheduling of scenes declared in the instantiation layer. Here, the scenes are encapsulated in an entity: the **plot**. This overcoat specifies the elements necessary for the realization of the stage. The plot is defined by the tuple (sc, G, S_{sup}). Here $sc \in S$ is a scene of the instanciation layer, G is a set of constraints that imposes a value to the state vector of a specific actor, before the completion of the scene and S_{sup} is a set of successors plots.

This method presents two advantages. First, the design is based on three layers that helps modularity and reuse. The abstraction layer helps the designer to constructs generic entities (actants) and a generic way of combining them. The second one (instantiation layer) is devoted to the implementation of generic entities and supports modularity by multiple inheritance of actants into concrete actors and combinations of actors in a concrete situation of execution (so called a scene). Finally, the dynamic layer helps organizing those concrete scenes into execution schemes. This approach constitutes a great support for extensible systems and reuse of entities for modeling. The second contribution concerns the dynamic construction of the system's global finite state automata from behaviors. Here, the designer has only to describe atomic behaviors for actants. The automata executing the situations is constructed automatically.

3.2. An hybrid approach

The CITE model is an evolution of the model based on situations that takes into account the specificities of ubiquitous experience, but can still be improved through a hybrid approach constructed on the common problem addressed by both projects Link H/R and PLUG. In both cases, it is first question to acquire and analyze dynamically navigation data/behavior to extract relevant information on users and their interests (observation/analysis) and secondly to provide dynamic adaptation rules (or recommendations) for a particular purpose (leading the user to add products in cart on web sites, improve the visitor experience) with respect to the knowledge constructed on the user and context.

We wish here to combine both approaches (top-down and bottom-up) for building a high-level architecture based on a jointly developed model by designer and by a machine-learning process performed on real behavior of the user. The proposed hybrid approach consists in sharing the automatons generated from the abstraction layer with the PerAc architecture whose purpose is to analyse and modify the experience unfolding in parallel of the execution engine. Indeed, the designer will first represent the whole experience through the classical top-down approach (modeling each actant and situation within the abstraction layer and implements them in the implementation and dynamic layers). The execution engine will unfold narrative plots with

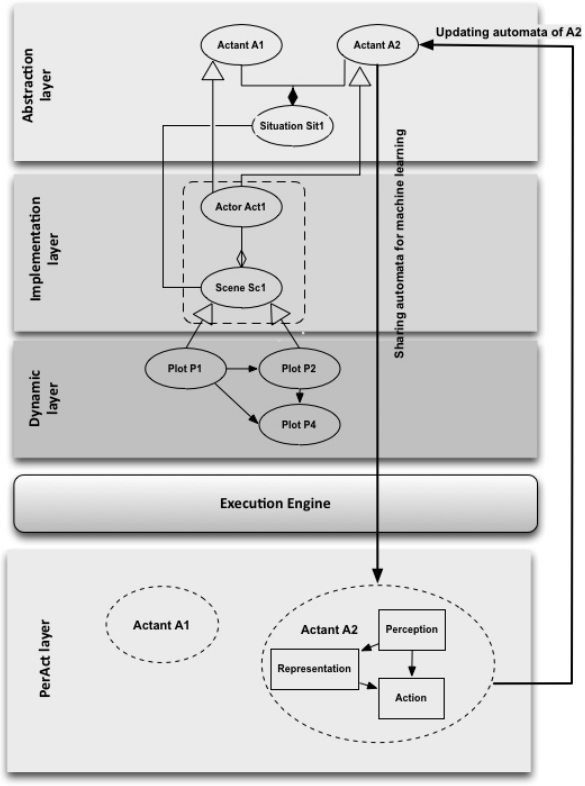

Figure 5. The proposed hybrid approach.

respect to the graph. Each actant implemented has its corresponding PerAc representation in the machine learning layer (the automaton is shared from the abstraction layer) and while executing, he analyses its own representation in order to perform improvements of its action automata. When the automaton has substantially changed, the ameliorated automaton is shared again with the abstraction layer and the whole experience model is recomputed. This pertinence cycle will be iterated during the whole experience execution.

3.3. Extensions: Time and spacEs

Such modeling requires to be extended in the context of ubiquitous applications. Indeed, if it is important to represent the interactions and content, the specificity of ubiquitous systems lies in the fact that users move in space to perform the actions and that time should be taken into account in a specific way as far as the session can be launched asynchronously and is often not limited in time.

Moreover, in this type of experiment the scalability is very important, i.e. the ability to add new contents or spaces can be difficult. Having a proper model and a writing tool for scripting and quickly integrate new elements to the experience is necessary. This promotes more the reproducibility of the experience in a new place with new contents.

Thus, the specificity of adaptation's approaches for ubiquitous and transmedia experiments, is that they passes through the following four layers: interaction, content, space and time. If there are models for traditional approaches, such that game play or web documentary ([19]), no narrative model solves problems specific to the transmedia or the ubiquitous systems.

To add the two dimensions related to time and space in the model, we will here propose some modifications to the model.

The spacE E. The dimension that concerns space is the easiest to add to the model. We wish to represent and easily modify the places in which the narrative situations unfold. Thus, the abstraction layer will need to embed a link between situations and places. Here, these are represented in a generic way and will eventually be implemented with the actual constraints in the implementation layer. For example, in the abstraction layer, a given space will be "outdoors" (which can be a constraint for the implementation of the action as it is necessary to know the GPS position of users). Every place of type garden, courtyard or street can implement this place and be used as a basis for the situation.

Moreover, this kind of mechanism is useful for setting up content that is, for the moment, a limit of our model. Indeed, currently, the contents are directly represented in the interactions associated with each actant (through variables). The reproducibility of the experience requires the ability to easily change the content (either exchange or simply modify). Another aspect is that you can easily use a simple interaction sequence in another context (for example, ask a question and expect an answer). It is therefore necessary to reduce the encapsulation of the content in the actants to create a mapping between the interactions and the externalized contents that will be easily editable by the designer.

The time T. The structure of the representation model facilitates the insertion of temporal constraints in the various layers. We can consider several levels for this time representation.

- time for the actor: we need here to represent the duration and constraints on actors's actions. For that purpose, the time management consists in adding in each actor a clock representing time of the actor's execution and the clocks required to represent the time for each interaction. These actors can be users or interactive entities of the experiment.

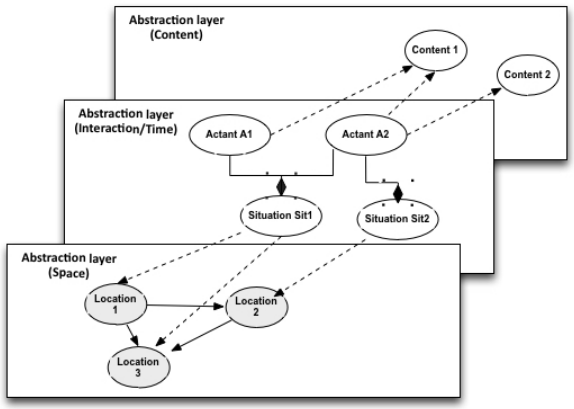

Figure 6. C, I, And E dimensions in the architecture for abstraction layer

Practically, since the transitions representing the actions of these entities are an extension of I/O automatons, we only need to add the elements of time management inspired from timed automata (clock, guard, invariant) on interactions representation. The situation automaton resulting from the presence of the actors (with timed interactions) in that situation and generated by the synchronous product will therefore be a timed automaton.

- the time of the situation: the presence of the actors (and their time constraints in the situation) induces de facto a timed management of the situation and we need to include a mechanism to follow the overall time in the situation. Then, an additional clock to monitor the time f owing in the situation and an invariant on this time is necessary. This invariant will guarantee, for example, that the situation will execute no more than a specifi duration.

- the overall time: the last layer of the model on time, will be the one that integrates the temporal constraints associated with the sequence of scenes. The same semantics of timed automata can be used to represent time management on the whole experience. Indeed, the challenge here is to add a global clock that will represent the time of the whole experience and constrain the plot graph described within the dynamic layer. This plot graph will then be a timed automaton and will pass through plots with respect to guards and invariants on the global clock.

4. Conclusion

In this article, we were interested in developing a generic model easing the conception of software or robotic applications that could be deployed into museums. Thanks to our previous experience in the conception of interactions with robots and a dancer (LINK H/R) or gamification in a museum (PLUG), we proposed a model, called "CITE" (stands for Content-Interaction-Time-spacE), which is based on a hybrid bottom-up/top-down approach.

To this point, only the CI (Content-Interaction) part of the architecture has been developed and tested. In particular, a first experiment embedding this model in a robot has been performed at the "Muséeum d'histoire naturelle" of La Rochelle in collaboration with researchers interested in marketing. The idea was to analyze how the introduction of a robot into a museum interfere with the visitors representation of knowledge, especially with young populations (teenagers).

Concerning the TE part, it should be developed in the future: we intend to create a new experiment in which two museums and an interdisciplinary team (marketing, computer science, robotics, anthropology, ergonomy) are involved. The interest of the CITE architecture would be to facilitate both the design of an application by non specialists (marketing, anthropology) and the adaptation to several places and contexts (two museums with different collections).

References

[1] J.S. Albus. Outline for a theory of intelligence. *IEEE Transactions on Systems, Man, and Cybernetics*, 21(3):473–509, 1991.

[2] P Andry, P Gaussier, S Moga, J P Banquet, and J Nadel. Learning and communication via imitation : an autonomous robot perspective. *Most*.

[3] R C Arkin. *Behavior based robotics*. MIT press, 1998.

[4] R Brooks. Elephants don't play chess. *Robotics and Autonomous Systems*, 6(1-2):3–15, 6 1990.

[5] P Gaussier, A Revel, J P Banquet, and V Babeau. From view cells and place cells to cognitive map learning: processing stages of the hippocampal system. *to appear in Biological Cybernetics*, 2001.

[6] Philippe Gaussier and S. Zrehen. PerAc: a neural architecture to control artificia animals. *Robotics and Autonomous Systems*, 16(2-4):291–320, 1995.

[7] C Giovannangeli and P Gaussier. Learning to navigate, progress and self-evaluation. *Symposium A Quarterly Journal In Modern Foreign Literatures*, 12(Cnrs 8051):8051–8051, 2005.

[8] A. J. Greimas. Séman tique structurale, 1966.

[9] Dilsun K Kaynar, Nancy Lynch, Roberto Segala, and Dipartimento Informatica. The Theory of Timed I / O Automata. 2005.

[10] M Maillard, O Gapenne, L Hafemeister, and P Gaussier. Perception as a dynamical sensori-motor attraction basin. *Advances in Artificial Life*, L(Ecal):37–46, 2005.

[11] J Nadel, C Guérini, A Pezé, and C Rivet. The evolving nature of imitation as a format for communication. In *Imitation in infancy*, pages 209–234. 1999.

[12] N. Rempulski, A. Prigent, and P. Estraillier. Supervision dynamique pour la scénarisation adaptative Modélisation et contrôle. In *AFADL 2010 : Approches formelles dans l'assistance au développement logiciel*, 2010.

[13] Nicolas Rempulski. Synthèse dynamique de superviseur pour l'exécution adaptative d'applications interactives. 2013.

[14] A. Revel. From Robots to Web-Agents: Building Cognitive Software Agents for Web-Information Retrieval by Taking Inspiration from Experience in Robotics. In *The 2005 IEEE/WIC/ACM International Conference on Web Intelligence (WI'05)*, pages 434–437. IEEE.

[15] A. Revel and P Andry. Emergence of structured interactions: From a theoretical model to pragmatic robotics. *Neural Networks*, 22(2):116–125, 2009.

[16] Arnaud Revel and Jacqueline Nadel. How to build an imitator. In Chrystopher L. Nehaniv and Kerstin Dautenhahn, editors, *Imitation and Social Learning in Robots, Humans and Animals*, pages 279–300. Cambridge University Press, Cambridge, 2006.

[17] Luc Steels. When are robots intelligent autonomous agents? *Robotics and Autonomous systems*, 15(1-2):3–9, 1995.

[18] J Stewart. The Implication for Understanding High-level Cognition of a Grounding in Elementary Adaptive Systems. *Robotics and Autonomous Systems*, 16(2-4):107–116, 1995.

[19] N. Szilas. IDtension–Highly Interactive Drama. *Proceedings of the Fourth Artificial Intelligence and Interactive Digital Entertainment Conference*, pages 224–225, 2008.

[20] Mark R Tuttle and Nancy A Lynch. Hierarchical Correctness Proofs for Distributed Algorithms. 1987.

[21] F J Varela, H R Maturana, and R Uribe. Autopoiesis: The Organization of Living Systems. *BioSystems*, 5(4):187–196, 1974.