Proceedings of the 2013 Federated Conference on
Computer Science and Information Systems pp. 1041–1046

Agent-based Architecture and Situation-based Scenario for Consistency Management

Thao Phuong Pham
L3i Laboratory
University of La Rochelle, France
Email: phuong-thao.pham@univ-lr.fr

Mourad Rabah
L3i Laboratory
University of La Rochelle, France
Email: mourad.rabah@univ-lr.fr

Pascal Estraillier
L3i Laboratory
University of La Rochelle, France
Email: pascal.estraillier@univ-lr.fr

***Abstract*—During interactions, system actors may face up to misunderstandings when their local visions contain inconsistent data about a same fact. Misunderstandings in interaction are likely to reduce interactivity performances (deviation or deadlock) or even affect overall system behavior. In this paper, we present agent-based architecture and scenario-structuring approach to deal with such misunderstandings and consistency. It is based on the notion of "situation" that is an elementary building block dividing the interactions between actors into contextual scenes. This model not only supports the scenario execution, but the consistency management as well. In order to organize and control the interactions, a "situation" contextualizes system's actors' interaction and activity, and includes prevention and tolerances mechanisms to deal with the misunderstandings and their causes. We also simulation experimentation on an Online Distance Learning case study.**

***Keywords*—Interactive system; adaptation; misunderstanding; situation- based scenario; consistency management**

I. Introduction

IN INTERACTIVE SYSTEMS, as games and simulators, the users and the internal agents can modify systems content and progress in real time through input adjustments. The interactive systems may adapt the system execution not only to user's actions, but also to user's profile and behavior, making these systems adaptive... In order to perform the adaptativity, the system must capture users' behaviors from their interactions. Then, according to system's logic and designer's logic, the system adjusts its execution to what it perceives of user's logic. Du to user's actions unpredictability, the execution process of an interactive system is also not predictable.

One of the important problems in interactive system is the potential misunderstanding between the users and the system and more generally between system's actors, virtual or physical. If the system does not capture correctly or confuses user's actions, or if the users do not understand what the system expects, that may lead to an erroneous interpretation of their behavior and an erroneous adaptation of system execution. This misunderstanding may concern user-system interactions, but it can also appear in any kind of interaction between any system's actors. It can be due to the incomplete actors' data or the non-determinism of actors' behavior and cause the interaction deadlock or application failure.

In our recent works, [1], [2], we have defined the misunderstanding in interaction as: when two or more system's actors have incoherent data in their local visions about the same fact f and these data is used during their interaction, that can cause an interaction deviation from the planned scenario. An actor may be human user or virtual system's agent. The local vision is actor's own knowledge about its external world (virtual environment, system's resources...), its relations with the others actors (subset of their states) and its own profile (internal state). So, our work focuses on the management of the consistency between the actor's behaviour logic and the system's logic and the consistency between the actor's local visions in order to handle the potential misunderstandings in interactions.

To handle misunderstandings we propose to contextually structure the application execution into interaction sequences called "situations" and including misunderstanding prevention and tolerance mechanisms. Each situation corresponds to a contextual resource-centered sequence of activities and events and is characterized by preconditions and postconditions. That allows the system to control the execution and to establish the casual links between the situations. This model confines actor's interactions in a given context in order to control them and manage the execution consistency. The consistency handling mechanisms, are inspired by techniques from dependability domain since there is an analogy between the misunderstandings in interactive systems and the errors handled in fault tolerant systems [7].

II. Misunderstanding in Interactions and Related Work

In the recent research, we can find several works dealing with the user-system dialogue where the communication is done through a real human language [3]–[5]. According to Rapaport [5], negotiation is the key to understanding. A cognitive agent understands by negotiating with the interlocutor or by hypothesizing the meaning of an unknown word from the context... A cognitive agent can negotiate with itself about something external by comparing it perception and internal knowledge in order to change or correct its own misunderstandings. Other works propose to use confidence scores to measure the reliability of each word in a recognized sentence [6]. Besides, Lopez-Cozar proposed to implement a frame correction module, which is independent of speech recognizer [4]. This module corrects misunderstandings in a sentence, caused by the errors in speech recognition, by replacing the incorrect frame with an adequate one. Karsenty and Botherel applied the adaptable and adaptive transparency strategies to TRAVELS project with the goal of helping the users to understand and

978-1-4673-4471-5/$25.00 © 2013, IEEE

react appropriately to system rejections and misunderstandings [3]. The ability of making system's interpretations explicit and informing the users on how to correct misunderstandings are two ways to help users handle them. This strategy is very effective in misunderstanding detection and raises the rate of appropriate user responses after system rejections. All of these works deal with the problem in speech dialogue where the misunderstandings are the more frequent. But the misunderstanding can be found in other forms of interaction like actions, gesture...

Our purpose is to define how we can treat the misunderstandings between the actors themselves, besides the user-system misunderstandings. It is not easy to recognize such class of misunderstandings. In the dependability domain [7], we find the inconsistency problem between systems and operators. The "automation surprise" is inconsistency error occurring when the system behaves differently than its operators expect [8]. It may be due to a mismatch between the actual system behavior and the operator's mental model of that behavior [9], and it can lead to "confusion mode" and sometimes critical failures. In general, misunderstandings come from the gap between user's logic and designer's logic, all along action planning between the actors. Many works, particularly in interactive storytelling, have been done to solve the mismatch between users' behaviors and system logic [10]–[12] by predicting the user's future actions and detecting the invalid ones that deviate the execution from the planned objectives. In general, prediction approach cost is very expensive, such as a short-term player behavior modeling module implanted in [10] to simulate how the world would change to player's actions. Moreover, this approach seems not well suited to a real-time interactive systems, nor to systems in which user's behaviors cannot be modeled easily by a set of rules.

Our approach will focus on software and component design model to integrate simple prevention and treatment mechanisms. Our solution relies on three points. *First*, we build robust agent-based architecture with specific additional components in charge of misunderstandings in interaction management. *Second*, we organize the actors' interactions as a situation-based scenario to facilate the interaction control. *Third*, we integrate into situations' dynamic execution the consistency management, including data synchronization, misunderstanding detection and treatment inspired and adapted from fault-tolerance techniques. These mechanisms do not try to predict users' behavior but will take into account users' state to adapt the system execution in order to avoid misunderstandings between actors. The system observes and analyzes users' states, detect the misunderstandings or their consequences and act to keep the consistency between actors' logics at the beginning and at the end of interaction sequences.

III. Agent-based General Architecture for Interactive Adaptive System

Several architecture models for interactive systems have been proposed according to the specific purpose of each work. We chose the approach of multi-agent system in [16] as a starting point to build our model. The advantage of this approach is that each agent can be organized and work autonomously and strategically. We added a special agent

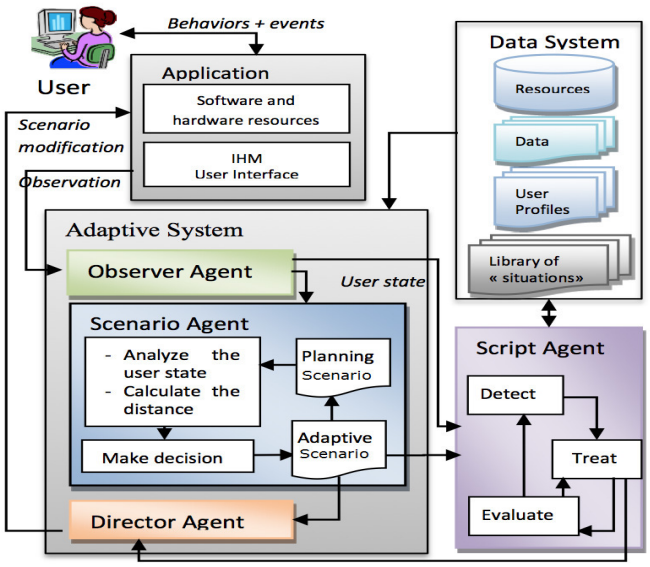


Fig. 1: General agent-based architecture for interactive system

called script agent besides the adaptation unit to manage the consistency. Figure 1 shows our overall architecture.

Observer agent: It observes user's behaviors and state, formalizes, normalizes and transfers them to the scenario agent.

Scenario agent: It makes decisions about scenario orientation according to user's state, planned scenario and permanent objective defined by the designer. This agent tries to find the best way to orientate the application execution. Scenario agent takes charge of a library of "situations" planned by the designer. These "situations" (defined in section IV) represent scenario components and are the interaction and the activity sequences that can take place in the application as, for instance, all possible scenes in a theater play.

Director agent: This agent receives the decision taken by the scenario agent. He takes in charge the production of the adaptive scenario and realizes a modification, an answer or an action adapted to the users.

Script agent: Its task is to track inconsistency in 3 steps:

- *Detection*: Detect, confine or partition the inconsistency between situation's actors in order to identify the causes of the misunderstanding.
- *Treatment*: Apply the handling mechanism or strategy to remove the inconsistency and to correct the deflected state that causes the incoherence.
- *Evaluation*: Estimate the efficiency of the treatments in order to improve the applied mechanism for the next time.

IV. Situation-based scenario

A. Approach of interactive storytelling

Our proposition is inspired from the interactive storytelling domain that focuses on scenario execution management.

Interactive storytelling is the unfolding of a story that the player's decisions impact [13], [17]. It also refers to how to generate stories which are both interesting and coherent. We consider that the interactions in an interactive application can be organized, strongly or weakly, as a story scenario. That allows us to adapt ideas from storytelling domain to organizing the interactions.

The scenario in interactive storytelling is represented by a series of actions/events linked together by cause and effect as in [14] or by ordered link as in [10], [15] or by Hierarchical Task Network planning as in [12] where each task is decomposed into subtasks until the primitive actions. But all of these scenario structurings are not suited to built complex interaction sequences where the user's actions are free, non predictable and depending on a great amount of context data. Hence, we propose the notion of "situation" that can be seen as a scene encompassing not only interactions execution but also interactions management and resources use. The situations are the basic narrative elements that facilitate interactions' planning and management by characterizing, contextualizing and confining them.

B. Scenario organizing with situations

1) Situation model: The interactions are split into a set of situations. Each situation is a sequence of interactions between two or more actors in a precise context to achieve a predictive objective, as shows the figure 2. It is characterized by: the preconditions, the postconditions, a set of participating actors and a set of resources. Due to the fact that actors' behaviors, especially human behaviors, are not always precisely modeled, and due to the influence of external events, the progression of a situation can be considered as an execution and adaptation "black box" where the interactions are executed in a non-predictable way. Furthermore, the situation includes consistency management. Itrepresents a set of mechanisms devoted to the prevention, detection and treatment solutions, in order to redress and adjust situation's progression in spite of misunderstanding and inconsistency problems. Consistency management is carried out all along the situation progression

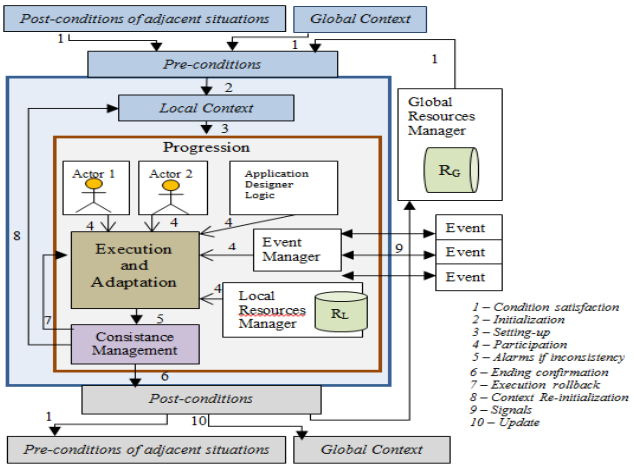


Fig. 2: Elementary Situation Structure

from the local context initialization to the post-condition completion.

2) Situation Graph and Application Execution: The situations are considered as the plot structuring elementary blocks. Each application provide a set of situations defining all the possible interaction sequences that can happen during the application execution. They can be grouped and linked together in order to build the overall application scenario. The scenario is then represented by a directed graph of situations. Each node is a situation and each edge is a transition from one situation to another. The situations graph shows the causal relationships between scenario situations. A scenario may have several beginnings and also some possible endings.

The situation-based scenario approach favorizes the execution control and interaction adaptation. The application progression becomes a scenario unfolding from one starting node to one final node on the predefined situation graph (it is taken in charge by the *scenario agent* in the global architecture). When there is more than one possible situation, the most pertinent one will be chosen by the scenario agent. To increase the adaptability, we can avoid the definition of a predefined graph. In that case, the situation choice is made according to the pre-conditions that best satisfy the global state and decision criteria. This method is flexible, adaptive, and applicable in "real time" during application execution, but it can lead to uncontrollable situation order or infinite loop, if the post-conditions and pre-conditions do not contain sufficient data.

V. Consistency Management Model within Situations

A. Handling Mechanisms

The *consistency management* that we propose consists of a set of specific methods, techniques and mechanisms that aim to handle the misunderstanding problem and to obtain data consistency all along the interactions. They are similar to the dependability techniques [7].

1) Prevention mechanisms: try to suppress misunderstandings occurrence conditions in order to avoid misunderstandings. To avoid data inconsistency, the proposed technique is the explicit declaration of all shared data before situation's interaction sequence start. It aims to identify and share actors' local visions in order to decrease the possibility of interaction deviation. Once the actors have collected the necessary data, they can start the interactions. The data synchronization is another method intending to compare the actors' local visions at a given moment during the interaction sequence in order to avoid the inconsistency of new perceived data. The synchronization can delay the interactions, so it should be done fast and not too frequently to disturb them as less as possible.

2) Tolerance mechanisms: aim to assure interaction continuation despite misunderstanding occurrence via misunderstanding detection and interaction recovery.

Detection: regular check of i) the shared data used during the interactions and ii) the deviation between actors' logics.

Recovery: once a misunderstanding is detected, the system apply one or several of the following techniques : **rollback**

- bringing the system back to a stable state, exempt from misunderstanding, to retry the interactions; **rollforward** - bringing the system to a new misunderstanding free state from which the interactions can go on; **reinforcement** - requiring from one or from all participant actors to do some additional interactions.

3) Removal mechanisms: involve misunderstanding detection and correction, followed by **reinitialisation** of the concerned interaction sequence, or of the whole execution process. The detected misunderstandings will be diagnosed to determine their causes: which data are inconsistent? Which ambiguities exist in the interaction context? Are there protocol faults? After that, an appropriate correction method will be applied to eliminate the related misunderstanding. Finally, the interactions have to be restarted from the last stable point or from the beginning.

B. Inclusion into the Situation Structure

Our situation-based architecture allows the integration of misunderstanding management mechanisms inside the situation in order to control the misunderstandings and their consequences all along situation execution. We define three phases (figure 3).

a) Prologue phase: The explicit declarations of interacting content and data are performed, to synchronize actors' local visions before they start to interact. If the initial data of all actors are identical from the beginning, the possibility of misunderstanding is reduced. If the inconsistency exists, a negotiation step will be performed between the inconsistent actors. Then, one or several of them will modify its/their data, or the divergent data will be isolated/removed and not considered during the interactions.

b) Interaction or Dialogue phase: when the interactions are carried out, the actors will update their local data step by step, as they continuously observe and perceive each other. Despite the initial local vision agreement, misunderstanding may nevertheless occur during the interactions. This is why their local knowledge is synchronized all along the interaction sequence in order to avoid that local data about same facts diverge in actors' local visions. One or several techniques of *reinforcement, rollback, rollforward* can be alternatively used.

c) Epilogue phase: All the interactions are done in the previous phase. If the post-conditions are fulfilled, we can exit the situation with the expected results. But if, for some reason, we do not reach the expected post-condition, the **script agent** has to detect and settle the existing incoherency in order to avoid the propagation of the misunderstandings to other situations. The system may also require actors to do some reinforcing interactions, or if necessary, make a rollback to a last stable state (in this case there must be systematic state saving mechanism), or, even a restart of the whole situation. The main goal of this phase is to quit the situation with the appropriate post-conditions and without latent or active misunderstanding. But, the rollback or reinforcing interactions may not lead the actors towards the planned post-conditions. Therefore, we add in the situation model a special exit point called "exception" that allows the current situation to be stopped at anytime without expected post-conditions and that leads to **exception handling situations**.

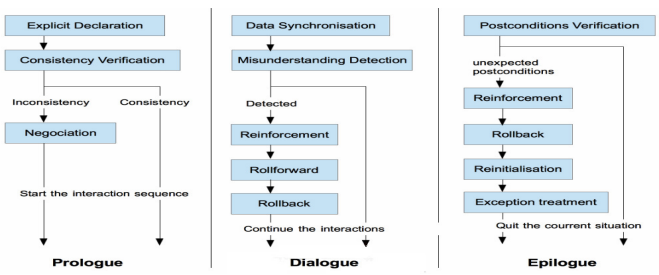


Fig. 3: Consistency management mechanisms

VI. Experiment on Online Distance Learning Case Study

To validate our approach we applied our situation-based methodology in our current online distance learning (ODL) project [2]. The project is devoted to the development of an online distributed platform that simulates a real classroom: teachers and learners carry out learning sessions as in a real life but by interacting through a virtual class environment. The platform integrates an interactive numeric board, camera, microphone and pedagogic tools (as file sharing system or virtual notebook) to support the courses...The figure 4 shows an example of courses scenario based on 6 situations. However, the users may face many difficulties: class supervision, course quality assessment, misunderstandings due to the weak system's interfaces and mechanisms to catch and manage user behaviors. The interactions between the actors in ODL contains numerous factors that may lead to misunderstandings as: multi-meaning or implicit behaviors; supervision tools' observation and interpretation imperfection; system component failures; incomplete, missing, implicit or wrong consigns...

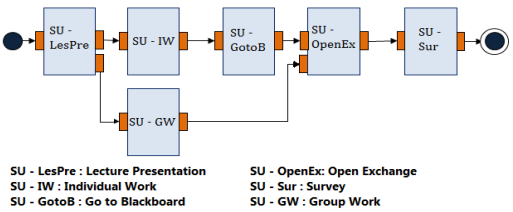


Fig. 4: Situation-based scenario example

A. "Individual Work" Situation Description

To deal with these various misunderstandings, we applied our situation-based solution including consistency management to a particular situation: "Individual Work" (SU - IW in figure 4). Each learner will work individually and has to do the exercises distributed by the system. The system provides additional exercises each time the learners send the previous exercises report. The expected post-condition is that all the learners reach a required knowledge level "*MaxKnowledge*".

Because of the long test duration and development for the real platform prototype, we chose to experiment our misunderstanding management mechanisms and agent-based architecture through a multi-agent simulation with the GAMA

platform[1]. We have 4 types of agents : "Teacher", "Learner", "Observer" and "ODL System". The Observer's role is to observe the state of sent exercises in order to evaluate learners' accumulated knowledge level. The distribution mechanism based on these observations and learners' skill level evaluations is taken in charge by "ODL System" agent that is a combinaison of 3 other agents in our model: scenario, script and director agent (figure 1). Potential misunderstandings in this situation occur when the system distributes the exercises that are incoherent given the learners' skill and expectation. They can result from wrong learners' exercise state observation or from inappropriate distributed exercise level. The misunderstanding handling is done inside the situation during its 3-phase progression (figure 5).

Prologue phase: The system checks each learner's connection status to begin the exercise series distribution.

Dialogue phase: In this situation, the interactions content refers to the exercises distribution and reporting. During learners' work, each observer agent supervises his associated learner's working state and his exercise report to collect data: partial or total termination, work duration, correctness rate. To avoid the wrong estimation of learner's skill and knowledge level, these data have to be synchronized between the observer and his learner after the exercise report is sent and before a new exercise is distributed.

Epilogue phase: To finish the situation the lerners must reach a given skill level after a given number of exercises. If a learner reaches this number without reaching the required skill level, the series will be stopped after a *session deadline* to avoid an abnormal long series. The system sends a *StopSignal* message to all learners to confirm the end of the exercise series after a predefined timeout. It refers to the exception treatment.

B. Experimentation Results

We run the simulation of "Individual Work" situation with the following parameters: 50 learners, 1 teacher, max knowledge level = 25, max difficulty level = 20, session deadline = 250 steps of simulation. We will measure a set of

[1]https://code.google.com/p/gama-platform/

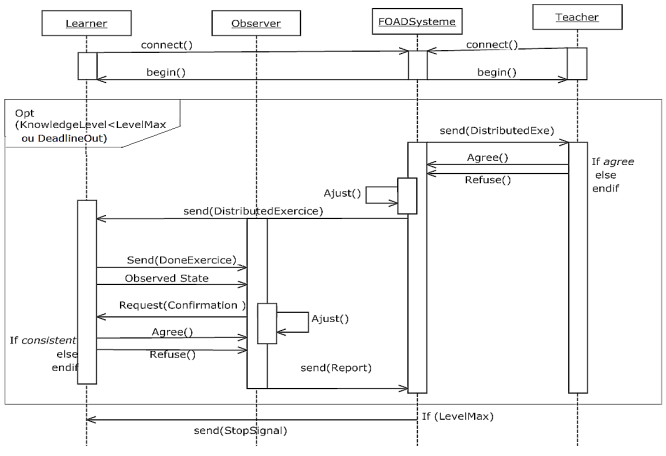
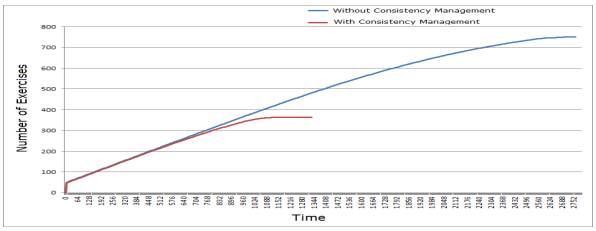


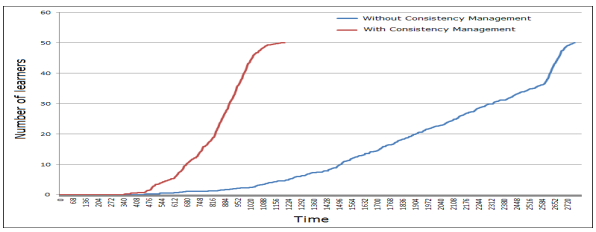
Fig. 5: Agents main interactions in the simulation

important factors influenced by potential misunderstandings:
N_e: total number of distributed exercises;
N_{notend}: total number of real non-finished exercises;
N_{bad}: number of bad observation by all observers;
N_{cor}: number of system observation corrections while detecting the wrong observed states (it refers to the synchroniaation times where consistency management is performed to remove inconherent data);
LI: learners' interest level that increases when the learners succeed and that decreases when they fail their exercises;
T_{total}: total session times (in *steps*) until the last learner has finished his series.

The data are recorded and calculated for the average values from 10 simulations lauching times in each measure. We compare these data between two cases: "with" and "without" the consistency management. The results are summarized in the table I. The total distributed exercises number N_e is twice more in "without" case compared to the "with" case. The average number of not finished exercises in "without" series is higher than in "with" series: 747.4 vs 363.4 also depicted in the figure 6(a). It is obvious that the session duration in "without" case is almost 2 times longer than in "with" case.



(a) Number of distributed exercices.



(b) Number of learners finishing exercise series.

Fig. 6: Comparason between 2 cases "With" et "Without".

The figure 6(b) shows the number of learners that have finished their whole series during the situation execution in the "with" and "without" consistency management cases. The lines shows that the learners work with more exercises and with longer duration T_{total} in the "without" case. We can make the same observation with the average measure values in table I.

Why do we have this difference result? When the consistency management is integrated in the situation execution to handle the potential misunderstandings, the observers have to adjust their observed data according to learners' "disagree" acknowledgments. Hence, the learner's skill level estimation will converge faster to the real value, and the difficulty level of the distributed exercises is more appropriate to his skill. The result is that learners can finish all the exercises and with higher correctness rate. In contrast, if no mechanism is added to control the inconsistency between learners and observers,

TABLE I: Statistical data comparason between 2 cases: With (Wi) et Without (Wo) the consistency management

	N_e		N_{notend}		N_{bad}		N_{obsnon}		N_{cor}		LI		T_{total}	
	Wi	Wo	Wi	Wo	Wi	Wo	Wi	Wo	Wi	Wo	Wi	Wo	Wi	Wo
1	330	735	23	64	83	103	93	114	83	0	78.98	66.1	988	2692
2	363	692	45	28	87	76	110	88	87	0	77.73	76.41	1104	2640
3	361	744	44	55	94	97	114	114	94	0	76.35	69.06	1076	2700
4	383	768	47	73	110	99	129	118	110	0	77.31	65.18	1160	2724
5	347	744	32	60	87	99	109	115	87	0	77.94	67.31	1024	2688
6	360	806	37	65	111	117	122	135	111	0	77.55	64.68	1108	2760
7	392	737	66	59	93	92	118	108	93	0	73.06	66.88	1188	2692
8	379	752	42	66	117	100	131	112	117	0	77.65	65.88	1140	2712
9	353	722	37	69	96	100	111	111	96	0	77.49	67.55	1048	2672
10	361	774	40	62	93	115	117	134	93	0	74.18	65.92	1084	2728
Ave.	363.4	747.4	42.4	60.1	97.1	99.8	115.9	114.9	17.8	0	76.82	67.71	1092	2641

a non-finished exercise can be perceived as finished, and vice versa. The skill estimation is less correct: higher or lower than the real one. There is a higher probability that the ODL system gives to the learners too difficult or too easy exercises. That delays the skill level progression and explains why the learners take more time to terminate the series.

VII. Conclusion

In this paper, we have presented the situation-based design methodology and consistency management mechanisms to handle the misunderstanding in interactions. Our approach is to contextualize the interactions between actors into "situations" and add to these basic narrative blocks consistency management mechanisms split into 3 steps: the *prologue*, data declaration and consistency verification, the *dialogue*, the interaction unfolding, local visions synchronization and misunderstanding treatment, and the *epilogue*, data update and agreement attainment. We do not seek to find a universal algorithm or a solution to deal with all types of misunderstandings in interaction. Our aim is to provide a management pattern that could be systematically used by the application designers or developers and that allow them to incorporate their own verification, synchronization, prevention and tolerance mechanisms adapted to the specific misunderstandings of their applications.

We have applied our methodology to a case study from an Online Distant Learning project. We have built a simulation of the "Individual Work" situation and integrated into it the proposed solutions to show how the consistency management operates on a simulation example. From the experimentation results, we have found out that our mechanisms reduce the incoherent data between learners and observers and improve the performance of exercise distribution: shorter session duration, lower exercise number, faster required level attainment... Even if the simulation is simple and does not cover exhaustively all the possible interactions that can occur in such situation, it illustrates the benefits of misunderstanding management during interaction progression.

The next step of our work is to perform the same experimentation and measures on the prototype under development with live case study and to check the relevance of our approach in other e-Learning "situations".

References

[1] P. T. Pham, M. Rabah and P. Estraillier, "Handling the Misunderstanding in Interactions : Definition and Solution," *in The Annual Int. Conf. on Software Engineering & Applications SEA 2011,* 2011, pp. 47–52.

[2] F. Trillaud, P. T. Pham, M. Rabah, P. Estraillier, and J. Malki, "Online Distant Learning Using Situation-based Scenario," *in the Int. Conf. on Computer Supported Education CSEDU2012,* 2012.

[3] L. Karsenty and V. Botherel, "Transparency strategies to help users handle system errors," *in Speech Communication*, vol. 45, no. 3, Mar. 2005, pp. 305–324.

[4] R. Lopez-cozar, Z. Callejas, N. Abalos, G. Espejo, and D. Griol, "Using Knowledge about Misunderstandings," *in Speech Communication,* 2010, pp. 523–530.

[5] W. J. Rapaport, "What Did You Mean by That? Misunderstanding, Negotiation, and Syntactic Semantics," *in Journal Minds and Machines,* 2000, pp. 397–427.

[6] H. Jiang, "Confidence Measures for Speech Recognition: A survey," *in Speech Communication*, vol. 45, no. 5, 2005, pp. 455–470.

[7] J. C. Laprie, B. Randell, C. Landwehr, and S. Member, "Basic Concepts and Taxonomy of Dependable and Secure Computing," *in IEEE Transactions on Dependable and Secure Computing,* vol. 1, no. 1, 2004, pp. 11–33.

[8] S. Combefis, P. S. Barbe, and C. Pecheur, "A Bisimulation-Based Approach to the Analysis of Human-Computer Interaction Categories and Subject Descriptors," *in EICS '09 Proceedings of the 1st ACM SIGCHI symposium on Engineering interactive computing systems,* 2009, pp. 101–110.

[9] G. G. King, "General Aviation Training for "Automation Surprise" ," *in International Journal of Professional Aviation Training & Testing Research,* vol. 5, no. 1, 2011.

[10] B. Magerko and J. E. Laird, "Mediating the Tension between Plot and Interaction," *in AAAI Workshop Series: Challenges in Game Artificial Intelligence,* 2004.

[11] H. Barber and D. Kudenko, "Generation of Dilemma-based Interactive Narratives with a Changeable Story Goal," *in the 2nd Int. Conf. on INtelligent TEchnologies for interactive enterTAINment,* 2008.

[12] R. Paul, D. Charles, M. McNeill, and D. McSherry, "Adaptive Storytelling and Story Repair in a Dynamic Environment," *in The Fourth Int. Conf. on Interactive Digital Storytelling ICIDS,* 2011.

[13] R. Champagnat, G. Delmas, and M. Augeraud, "A Storytelling Model For Educational Games : Heros Interactive Journey," *in International Journal of Technology Enhanced Learning 2,* 2010, pp. 4–20.

[14] B. Karlsson, A.E.M. Ciarlini, B. Feijo, and A.L. Furtado, "Applying a Plan-Recognition / Plan-Generation Paradigm to Interactive Storytelling," *in Workshop on AI Planning for Computer Games and Synthetic Characters,* 2006.

[15] A. Silva, G. Raimundo, and A. Paiva, "Tell me that bit again...' Bringing interactivity to a virtual storyteller," *in Int. Conf. on Virtual Storytelling,* 2003, pp. 1–10.

[16] K. Sahaba, P. Estraillier, and D. Lambert, "Interactive educational games for autistic children with agent-based system," *in 4th Int. Conf. on Entertainment Computing (ICEC'05),* 2005, pp. 422–432.

[17] J. Lebowitz and C. Klug, "Interactive Storytelling for Video Games: A Player-Centered Approach for Creating Memorable Character and Stories", Focal Press, 2011