Control structure for an immersive mixed reality environment

Marija Nakevska, Jun Hu, Geert Langereis and Matthias Rauterberg*

Department of Industrial Design, Eindhoven University of Technology

ABSTRACT

Immersive mixed reality environments offer new possibilities to reproduce reality or embodied presence with constructing elaborate fantasy worlds and provide to the user an intense and seemingly real experience. Mixed reality gives possibilities to create deeply dimensional narratives and simulations that put the user in the center of the action. In this work we describe an interactive mixed reality installation named Alice, consisting of six separate stages based on the narrative 'Alice's Adventures in Wonderland'. To be able to achieve the intended experience we have to build complex and heterogeneous distributed system, composed of sensors, actuators, virtual reality, application components and variety of processing components that manage the flow of context information between the sensors/actuators and applications. The main design problems of this system are in designing and specifying the overall system structure. We examine the possibilities of structural organization including gross organization and global control structure; protocols for communication, synchronization and data access; scaling and performance and selection among design alternatives.

Keywords: system architecture, immersive mixed reality, interactive storytelling.

1 INTRODUCTION

Immersive mixed reality environments are using a large palette of techniques to create fantasy worlds, including various multisensory effects like various forms of tactile sensations, fog, movement of a seat or floor, and artificially produced smells. All the components of other engaging forms of digital entertainment can be included; as sound and moving images, computer-generated intelligent characters, three-dimensional simulations, embodied characters and new types of spaces.

We are exploring a new application of mixed reality for a novel direction in human-computer interaction named 'cultural computing'.[1] The Alice project aims to create an interactive installation based on the narrative of 'Alice's Adventures in Wonderland' that encourages people in Western culture to reflect on themselves, addressing issues such as logic, rationality, and self.[2][3] The Alice installation, located at the Eindhoven University of Technology, is one of the biggest mixed reality installations of this type. Similar work, using digitally created elements and real physical object contained within the same space, can be found in museums, mixed reality in military scenarios, large screen immersiveness for audiences, fulldome productions and interactive theme park rides. Experiences provided with theme park rides are extremely exciting but are rather passive, since the theme parks need throughput of visitors and the interaction is not individual. In addition, theme park rides normally do not address the visitors cultural, nor self reflective level. One interactive storytelling theme park ride is The Buzz Lightyear AstroBlaster, build around the one of the characters of Toy Story films [4]. The University of New Mexico works on the adaptation of children's book 'Arrow to the Sun' using fulldome immersive entertainment. The 'ZENetic Computer' is an installation that incorporates certain elements of the Japanese Zen culture. Tosa et al. (2005) projected this style of communication into an Eastern sansui world as an interactive system with which the user can create a virtual world by manipulating 3D images of sansui paintings [5][6].

Mixed reality offers a lot of creative opportunities, but the complexity and cost of involved technology hinders the process of utilizing this form of interactive experience. The design includes implementation of a distributed system consisting of autonomous components that need to collaborate. The type of components of the system can vary in a range from high-performance mainframe computers to small sensor nodes. We are examining one of the most important aspects of implementing an immersive mixed reality installation: the system architecture. The impact of the architectural choices extends from the proper event management during operation to the integration standardization constraining the development team. Indirectly, it will affect the cultural experience we can facilitate with the installation. When formalizing the architecture, we take both formal and spatial elements of the system and interprocess communication and synchronization into account. Furthermore we are presenting the Alice project with the challenges of the hardware and software design to provide the intended experience, considering the integration at the overall system level.

2 CONTROL STRUCTURES

2.1 Behavioral structures

We are interested in context aware interaction, using an interactive narrative as a new medium in digital entertainment in which users create or influence a dramatic storyline through actions. The storyline takes the agency of controlling the flow of events but the user is able to change it and influence the behavior of the system.

These two-limit cases lead to different behavior paradigms, one depending on the events and actions related with the visitor and second pre-defined approach, autonomous behavior of the system. General idea of the plot is created and the autonomous components of the system should support the simulation dynamically evolving around this plot. This is hybrid approach between two opposed approaches, supports a scenario with predefined action points but also recognizes changes in the state and reacts on it. The scenario is specified prior to simulation and the installation does not anticipate changes in the plan at run-time. The autonomous system more or less follows a predefined plot but is not able to react on all of the unexpected events. The reason is that such installations consist of stages which are work in progress and so not all possible user actions can be anticipated. To be able to react on critical unexpected events, we should anticipate the

* {m.nakevska,j.hu,g.r.langereis,g.w.m.rauterberg}@tue.nl

possibility of real-time external interventions in order to finalize experiments smoothly. In addition, due to safety reasons our system has to implement surveillance cameras with video stream and to support manual and remote control of specific critical points in the installation. As a result, the architecture should comprise tracks for (1) the pre-defined plot, (2) smooth intervention at unexpected events, and (3) surveillance. In addition, the system should be able to store events for research purposes.

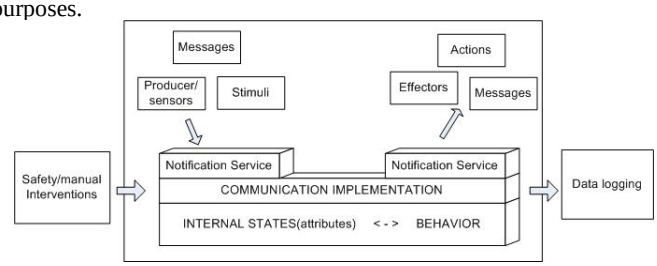

Figure 1: Schematic view of the affecters in the system.

We consider an interactive narrative, relying on a pre-defined scenario with various conditions; every stage has an intended goal or a plan of actions to achieve at a certain moment. The scenario of every stage or sub-story is described as a stack of intermediate goals or tasks to perform. To simulate certain scenario we have to distinguish three main components: a perception module responsible of sensing the world, an action module responsible of responding to the sensory stimuli and a decision module responsible of analyzing and reasoning (see Figure 1). An additional requirement is the observation and logging of the data produced from the events (sensors/actuation) and video surveillance.

The perception consists of creating an information flow from the environment to the system and gives knowledge of the surrounding environment. The environment is perceived through visual, tactile, haptic and auditory sensors, it includes physical mechanisms such as cameras, detectors or sensors.

In immersive environment the real world representing interactive narrative consists from complex events which are hard to be reduced to computational resources. In many scenarios, sensing the world by analyzing and extracting valuable information from raw data is a time consuming process. The creation of this kind of interactive environment is restricted by defining events and fitting the requirements in computable expression. We define event as a happening of interest that can be observed within the system.

The events we want to define in this environment are:

- Physical events such as appearance of a person detected by sensor or performing of a specified interaction with the system
- Action events, as pre-programmed behavior of the system and manual interventions
- Arbitrary detectable state change in a computer system
- Timer event that indicates progression of real time

In order to build a storyline we have to predict the events in the system and program the behavior of the system and also predict how the participant would move through the system.

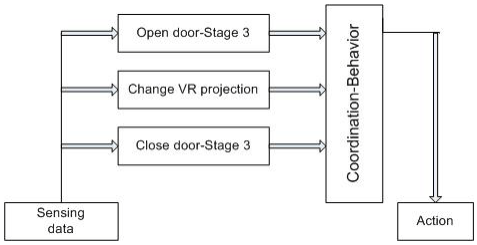

Figure 2: Selection of an action based on sensing data and coordinated global behavior of the system.

Actions are translated into concrete events, including modifying the immediate environment, the state of the system and to communicate results to other entities in the system. When an action is selected (see Figure 2) and performed its invocation alters the environment, influencing the selection of future actions. Each component focuses on the local behavior, but not on the behavior of the global system. In order to obtain a consistent global behavior the system has to provide support for the interactions between entities themselves and with their environment.

2.2 Architectural structures

To organize the components of the system properly, we have to define the structural elements of the system and the architecture determining the interaction and placement of the individual components. We are examining the main types of network paradigms used in distributed systems: centralized, decentralized, hybrid and middleware system architectures.

Centralized systems manage the complexity of distributed systems in terms of clients that request services from servers. Clients are sending orders to the relevant central server and receive relevant information back. The model of client-server architectures can be flat (Figure 3(a)) or hierarchical (Figure 3(b)) depending if clients communicate with a single server or in hierarchical model the servers in one level are acting as clients to higher level servers.

A centralized system is very sensitive to changes, and requires more bandwidth, because of the communication bottleneck towards the server of the structure. Hierarchical topology helps the flexibility of the system, and creates richer possibilities for implementation of fault-tolerance techniques [7] [8] [9].

Decentralized system or peer-to-peer architecture (Figure 3(c)), take advantage of resources, services and content offered by the network [10] [11]. A decentralized system architecture increases robustness because it removes the single point of failure that is inherent to a client-server based system. In decentralized systems the coordination of processes is difficult to keep consistent with the global state. Decentralized coordination hinders the programmability of the global behavior of the system. Computing power and bandwidth can have an impact on overall performance, since all the nodes are not created equally.

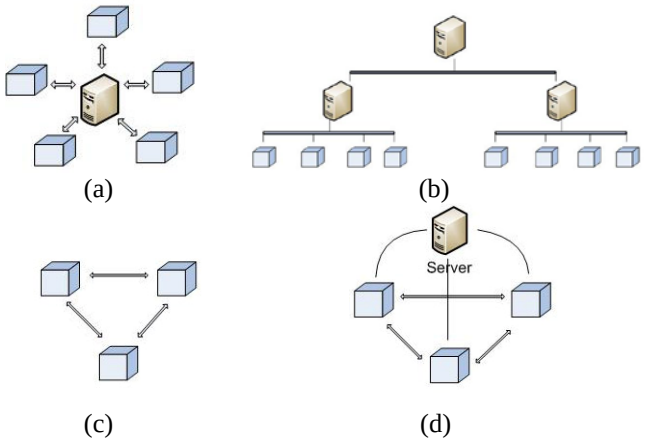

Figure 3: Centralized ((a) flat and (b) hierarchical), (c) peer-to-peer and (d) hybrid distributed systems.

Hybrid systems are using peer-to-peer systems which incorporate traces of client-server relationship (Figure 3(d)). The communication is done first with the server to obtain the

location/identity of peer, following by direct communication with the specific node. Hybrid hierarchical topology provides stability in requests transmission, low memory consumption and relatively low waiting time [12] [13] [14].
Middleware forms a layer between applications and distributed platforms, as shown in Figure 4, it provides a degree of distribution transparency, hiding the distribution of-data, processing, and control from applications. In event-based architectures processes essentially communicate through the propagation of events, event propagation generally is associated with publish/subscribe systems.

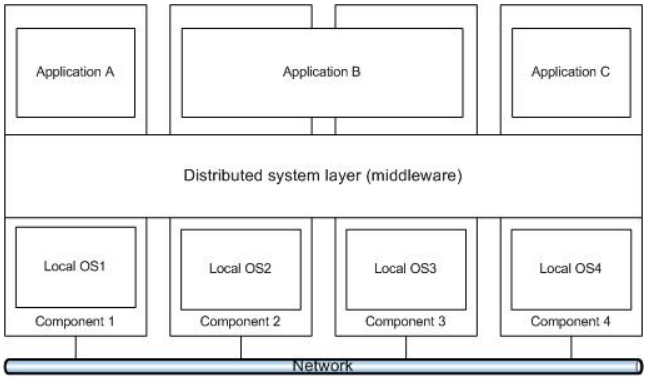

Figure 4: A distributed system organized as middleware. The middleware layer extends over multiple machines and offers each application the same interface.

Middleware systems are required to be easy to configure, adapt and customize. Middleware systems need to be general; applications have specific extra-functional requirements that conflict with aiming at fully achieving this transparency. The requirements for generality and specialization have resulted in middleware solutions which are highly flexible, but the price to pay for the flexibility, is complexity [15] [16] [17].

2.3 Communication and synchronization

Interprocess communication and synchronization is one of the most important issues in a distributed heterogeneous system. Communication in distributed systems is always based on low level message passing as offered by the underlying network. The most common data models used in communicating messages between systems are: name/value pairs, objects and semiconstructed data e.g. XML. We look in three widely-used models for communication: Remote-Procedure Call (RPC), Message-Oriented Middleware (MOM) and data streaming.
As we described, the complexity of the system, the components integrated in it, and the need of timing models and synchronization mechanisms have different tolerance level. Remote Procedure Call is a synchronous communication model, invokes external procedures that reside in a different system or machine in the network. An example of RPC systems is Open Network Computing(ONC) from Sun Microsystem (Sirinivasan, 1995). Distributed Computing Environment / Remote Procedure Call (DCE/RPC) is used as the basis for Microsoft COM+ middleware (Eddon and Eddon, 2000; Pinnock 1998) [18]. Jini is advanced service oriented architecture, similar to Java Remote Method Invocation, is a centralized model where the lookup server implements broker communication between client and service [19] [20].
The prevalent interaction in distributed system is the exchange of messages (direct communication) or events/updates reflecting environment changes. Direct communication between the components of the system leads to complex architecture and network overload. We are examining two types of efficient communication methods: message-passing and blackboards.

2.3.1 Message passing systems

A message passing system uses a straightforward subscription-based addressing scheme. Various middleware products are fulfilling this requirement and are characterized as messaging, message oriented middleware, message queuing or publish-subscribe. Existing software solutions are: Common Object Request Broker Architecture (CORBA), Event Service (Bolton 2001; OMG 2004), the Elvin Notification service (Segal and Arnold, 1997), the InfoBus data controllers (Sun Microsystem, Inc., 1999), iBus channels (SoftWired AG, 1998), Robot Operating System (Stanford Artificial Intelligence Lab.,2007), and ZeroMQ (iMatrix Corporation).
Systems in which communication is done through publish and subscribe paradigm require the sending agents (publishers) to publish messages without explicitly specifying recipients or having knowledge of intended recipients. Similarly, receiving agents (subscribers) must receive only those messages that the subscriber has registered as being interesting. This decoupling between senders and recipients is usually accomplished by an intervening entity between the publisher and the subscriber, which serves as a level of indirection. This intervening entry is a queue which is used to represent a subject or channel of communication. A subscriber subscribes to a queue by expressing interest in messages enqueued to that queue and by using a subject or content-based rule as a filter. A component may subscribe to any number of pre-existing channels and may also create and destroy channels at will. Messages can be sent to any channels [21][22].

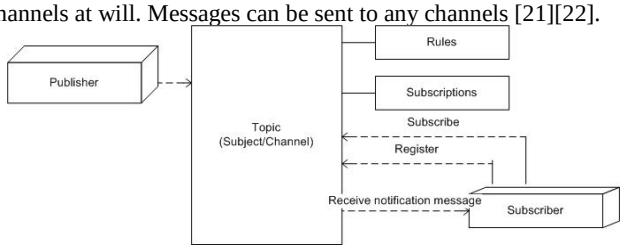

Figure 5: The publish-subscribe paradigm.

Message passing is intended as a lightweight mechanism for synchronizing behavior between multiple agents as well as for representing hierarchical command structures among components.

2.3.2 Blackboard systems

A blackboard system provides a shared memory framework that agents can use to store and exchange knowledge. In this architecture the components, contributors, communicate by updating the blackboard [23] [24]. It is analogous to a team of experts who communicate their ideas by writing them on a blackboard. A moderator object determinates the order in which contributors perform these updates. A blackboard is essentially a named collection of key-value pairs that has global visibility. Entities of the system can post values to and read values from any key on any blackboard. Blackboards are useful in simulations where groups of components of the system share a common situational picture or publish information without knowing the identity of the recipients.
Smart-M3 technology combines the ideas of distributed, networked systems and semantic web, enabling smart environments and linking real and virtual worlds [25].

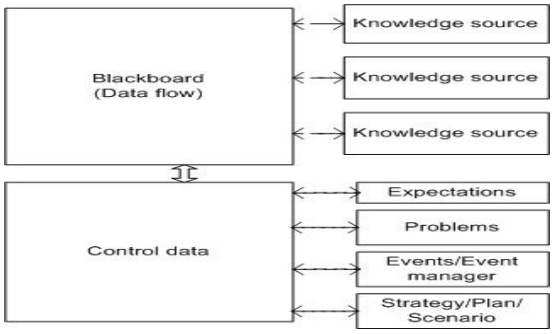

Figure 6: The Blackboard model.

2.3.3 Synchronization

The blackboard system provides a shared memory framework that agents can use to store and exchange knowledge. In this architecture the components, contributors, communicate by updating the blackboard. It is analogous to a team of experts who communicate their ideas by writing them on a blackboard. A moderator object determinates the order in which contributors perform these updates. A blackboard is essentially a named collection of key-value pairs that has global visibility. Entities of the system can post values to and read values from any key on any blackboard. Blackboards are useful in simulations where groups of components of the system share a common situational picture or publish information without knowing the identity of the recipients.

3 THE ALICE PROJECT

The ALICE installation consists of six consecutive stages based on the narrative 'Alice's Adventures in Wonderland'; each represents a chapter or part of it. The installation is built on two floors (the ground floor represented in Figure 7), and the designed experience is for one person at a time. From start to end the user undergoes immersion that engages the user in an experience closer to the author of the book has described for the main character Alice.

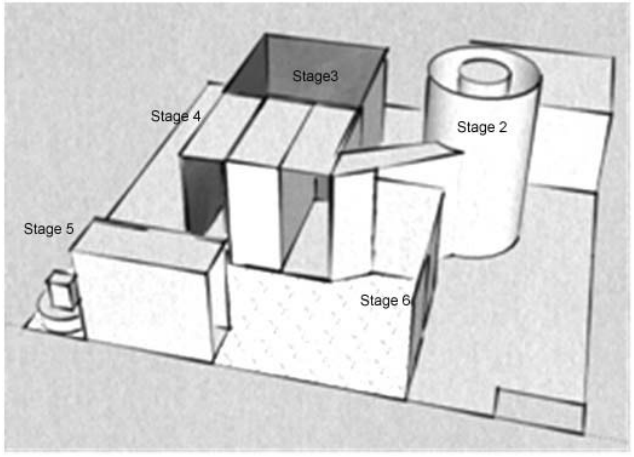

Figure 7: 3D model of the actual installation, presenting Stage 2, 3, 4, 5 and 6.

Following, we will explain the function of each part, summarizing the section from the original narrative, impression of the design of each stage and its technical implementation:

Stage-1: 'In the park'; In the first scene of Carroll's original book, Alice is bored and sleepy sitting on a bank with her sister. A white rabbit runs close by her and while looking at its pocket watch it cries out "Oh dear! Oh Dear! I shall be late!". We simulate natural environment in which the user is expected to wait until he/she is bored and thereby enters a similar state of mind as Alice in the original story. The visitor is surrounded with 360º panorama picture of Cambridge, location in nature at the river Cam. The room is supported with strong light behind the printed fabric which gives impression of brightness of a summer day. The visual simulation is supported by two virtual reality projections, also supported with appropriate audio simulation of natural environment. The appearance of the rabbit in the scene has a goal to seduce the visitor to follow with a catch-me-if-you-can behavior. To imitate the movement of the rabbit we are using virtual reality projection and physical actuation of artificial grass, successively synchronized, together with audio simulating the speech of the rabbit.

Stage-2: 'Down the Rabbit Hole'; Alice falls down the rabbit hole in an unusual fashion. The speed of her fall is very slow and she is able to interact with objects that are attached to the walls of the rabbit hole, such as cupboards and bookshelves; the fall takes such a long time that Alice engages in a conversation with herself. In the ALICE installation we are bridging the fall in the rabbit hole with a seat mounted on a rail. Once the visitor is safely seated it takes the visitor down the tunnel in a spiral movement (see Figure 8).

A securely controlled gate prevents the visitor to pass by the seat and thereby endangering him/her to truly fall down the rabbit hole. The seat is only activated once the armrests are put down, which prevents the visitor from falling off while moving. Infrared cameras are placed inside the rabbit hole to allow the experimenter to monitor the visitor. Along the walls of the tunnel, cupboards, bookshelves and lamps are mounted. The speed of the electric seat is slow.

Figure 8: Visitor entering the rabbit hole.

Stage-3: 'Shrinking and Growing'; Alice enters a dark corridor with many doors, which are all locked. She approaches a glass table on which a small golden key lays. She uses the key to open a tiny door that leads to a garden, but Alice is too tall to enter. She approaches the table again, and this time she notices a bottle labeled 'Drink Me' and later a little cake labeled 'Eat Me'. By drinking from the bottle, she shrinks and by eating the cake, she grows. Eventually, she manages to have the appropriate size to enter through the tiny door.

To be able to manipulate the relative size in comparison to the environment we are using a cubic CAVE. The walls and the ceiling are made of white semi-transparent material. We used the back-projection method to project a seamless virtual environment onto the walls of the CAVE. A sliding side is connected to the entrance and the exit tunnels, enabling a 5-side full projection when the visitor is in the CAVE (see Figure 9). Both the entrance and the exit tunnels are also build as a 3-side projection CAVE, but for the time being only the exit tunnel is used for the stage 4 (the pool of tears).

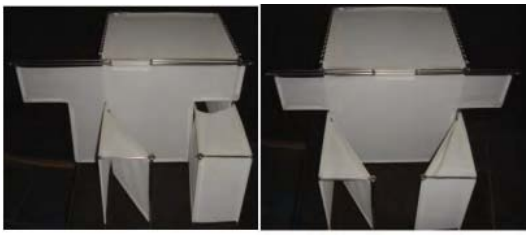

Figure 9: Model of the physical implementation of the 5-sided CAVE with the sliding doors.

The visitor entered the cave and had the impression to stand in a virtual room (Figure 10(a)). A cookies box labeled 'Eat Me' and a bottle labeled 'Drink Me' are placed on top of a small table (see Figure 10(b)). When the visitor drinks from the bottle, the virtual room enlarges, giving the impression that the visitor is shrinking. When eating the cookie, the virtual room shrinks, giving the visitor the impression that he/she is growing. On one side of the room, a door is shown. Once the visitor reaches the appropriate size, the wall on which the door is shown moves aside and thereby allows the visitor to enter stage 4.

Figure 10: (a)Virtual reality model of the projected room and (b)"Eat me, drink me" interaction equipped with sensors.

We used CryEngine 3[27] to synchronize the five projectors. The model of the virtual room was developed in Maya. The bottle features touch and tilt sensors to detect drinking. The cookie box is equipped with a microphone that allows us to detect the visitor's chewing sounds when eating the cookie.

Stage-4: "Pool of tears"; During her growing and shrinking experience, Alice cries many tears. When she walks through the tiny door she enters a pool of tears. She talks to a mouse that swims alongside her and together with some other animal they finally reach the shore. When the visitor enters stage 4, the sea with a mouse in it is projected on one of the walls. In addition, a smoke machine creates an impression of moisture and adds a mystical feeling to this stage. The visitor walks along the projection and thereby enters stage 5.

We created a virtual pool of tears with the use of back-projection and we also use fog machine to enhance the effect (Figure 11(a)). The animation of the mouse in the water was created in 3D Studio Max.

Stage-5:'Advice from a Caterpillar'; The state of confusion in the Alice character is emphasized in her conversation with the Caterpillar: 'Who are you?'. We created a robotic caterpillar to engage the visitor in a similar dialogue (Figure 11(b)). Microphones recorded the utterances of the visitor and a simple dialogue system manages the conversation. Since most of the questions are metaphysical or mystical, a conclusive dialogue can be created without an extensive Artificial Intelligence (AI) for the caterpillar.

Figure 11: (a) Pool of tears- projection and effect produced with smoke machine (b) A visitor is talking with the caterpillar, a robotic dialogue agent.

Stage-6: 'Talk with the Cheshire Cat'; The Cheshire cat involves Alice in a dialogue about logical reasoning and madness. During the dialogue the cat disappears at times completely and sometimes only its grin remains visible. We created a virtual Cheshire cat that is projected on a screen (Figure 12). When the visitor approaches the cat, it engages him/her in dialogue similar to the one in Carroll's book.

Figure 12: Left, different transformation of the Chashire cat, right visitor engaged in conversation.

A Microphone was hidden next to the screen to record the visitor's responses. A simple 2D animation tool (Adobe Director) is used for the animation of the cat and the dialogue management. Similar to the Caterpillar, most of the questions are metaphysical or mystical, and hence a conclusive dialogue can be created without an extensive Artificial Intelligence (AI).

3.1 Control structure

We described the Alice project with the six stages implemented in an immersive mixed reality installation; our goal is to use this automated system to bring a smooth and holistic experience to the visitor. To be able to achieve this, we have to control and structure the path through the storyline and in the same time create a free to explore interactive environment.

To be able to control the behavior of the system we have to anticipate the possible events and interventions in the system but also the communication service. The implemented storyline has linear and sequential flow of the stages and events; Figure 13 shows the sequence diagram of the events in Stage 2 and Stage 3. The system can be described as state machine which moves from state to another by means of action. A stage is required to implement the transitions commands from the central controller, and stage change notification to the central controller of the global behavior. Figure 14 depicts the possible state transitions in a stage.

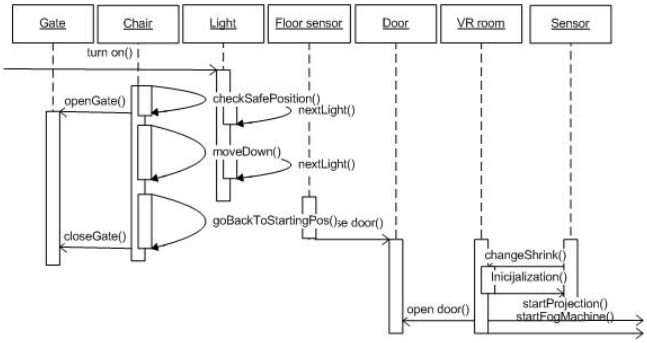

Figure 13: Sequential diagram of triggered events in Stage 2 and Stage 3 in the Alice project.

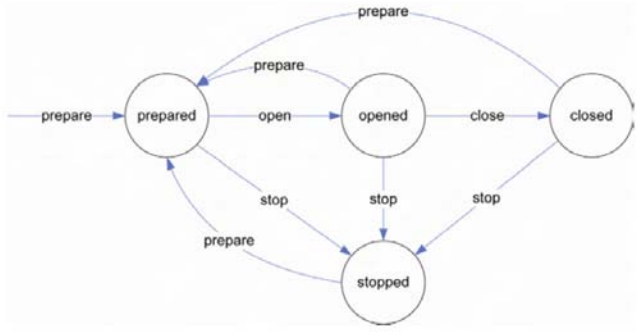

Figure 14: Stage state transitions.

In the Alice installation, every stage has its own requirements on the setup and employs a different set of technology. The stages are often composed of several processes executed in parallel and the separate entities may influence each other behavior. The time dependency of the elements influences the synchronization of the processes and actions. In multimedia time-depended elements are video, audio and animations; timing is crucial to this continuous data streams. For instance, the implementation of 5-sided CAVE in the Alice project employs advanced software technology [28] for synchronization of audio and video streaming and the rendering of the virtual reality. The computing, sensing and actuation hardware components needed for Stage 3 are shown on Figure 15. Except diversity of hardware components, the practical implementation brings more complexity because of the use of many programming languages, such as C++, Java, Python, JAL, and ActionScript.

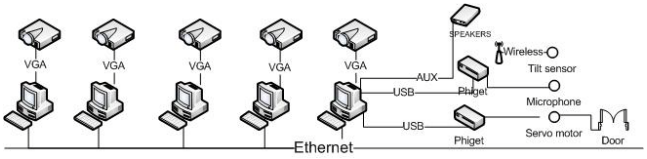

Figure 15: Hardware components needed for stage 3 in ALICE project.

In previous work we presented a multi-layered, multi-agent based structure based on the distributed architecture proposed by Hu (2006) [27], in which PAC (Presentation-Abstraction-Control) agents [28, 29] are distributed over a network and connected through so-called Channels(see Figure 16).

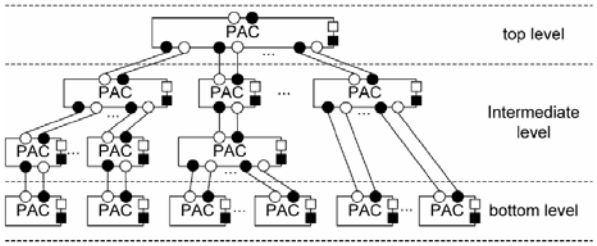

Figure 16: Distributed Presentation-Abstraction-Control over channel connections.

In this architecture a channel generalize the data and control communication among agents. Applying this structure to the Alice installation results in a three-layer structure. The channels between the bottom level and the intermediate level are mostly implemented over USB or serial cable connections or over Bluetooth radio. The channels between the central controller and the stage servers are implemented over TCP/IP network connections. An XML based communication protocol is designed and implemented directly on top of TCP/IP, to ensure the minimum requirement on the software packages or programming languages used by the stage servers.

This structure enables the ALICE project to decompose the complexity of the entire installation into loosely coupled stages, and to organize the project in a similar hierarchical manner that distributes the tasks in smaller teams and coordinates the progress at a higher level. To be able to coordinate the global behavior the system needs shared memory. In order to meet all this requirements we have to design hybrid system architecture integrating locally synchronized sub-stages of the system while the global behavior is controlled within shared memory and message-oriented middleware.

4 Conclusion

We have introduced and discussed an artistic and interactive installation that is used as a research platform for the new field named cultural computing. In this project we took advantage of novel technologies and mixed reality, to create rich experiences that influence the visitor. We are constructing in real-time a personalized narrative and flow of events. To maximize the potential for engaging the user in an interaction, we create immersive and interactive environment that reacts to his/her input and behaviors including motions, displacements and physical environments. Every stage uses one or more computers, connecting to different type of microcontrollers or hardware interfaces that are again connected to a variety of sensors and actuators. In this paper we described the problems and challenges in creating immersive mixed reality environment. We examine the possibilities of structural organization, implementation of behavior paradigms, depending on the events and actions related with the visitor and autonomous pre-defined approach. In order to obtain a consistent global behavior the system has to provide support for the interactions between entities, coordination between sensing data, pre-defined scenarios and selection of actions by the autonomous environment.

References

[1] M. Rauterberg. From Personal to Cultural Computing: How to Assess A Cultural Experience, in *uDayIV-Information nutzbar machen*, 2006, Pabst

[2] C. Bartneck, J. Hu, B. Salem, R. Cristescu and M. Rauterberg. Applying Virtual and Augmented Reality in Cultural Computing, in *The International Journal of Virtual Reality*, 2008, 7(2):11-18

[3] J. Hu, C. Bartneck, B. Salem and M. Rauterberg. ALICE's adventures in cultural computing, *Int. J. Arts and Technology*, Vol. 1, No. 1, 2008

[4] J. Garlington. Real-time interactive graphics. Taking Location-Based Entertainment to the Next Level - DisneyQuest and its Challenges for Computer Graphic Imaging, *Computer Graphics*, 1999.

[5] N. Tosa and S. Matsuoka. ZENetic Computer: Exploring Japanese Culture, *Leonardo*, vol. 39, no. 3, pp. 205-211, 2006.

[6] N. Tosa, S. Matsuoka and H. Thomas. Inter-culture Computing: ZENetic Computer, in *ACM SIGGRAPH 2004 Emerging technologies 2004*, Los Angeles, California: ACM

[7] A. S. Tanenbaum, M. van Steen. Distributed Systems - Principles and Paradigms. Secound Edition, 2006

[8] Z.Zhang, Z.Duan and Y.Thomas. On scalable design of bendwidth brokers, *IEICE TRANS.COMMUN/*, VOL.E84-B, NO.8 August 2001

[9] K. Vanthournout, G. Deconinck. The Hierarchical-distributed Topology for the communication infrastructure of complex embedded automated systems. *3rd Int.Workshop on Design of Reliable Communication Networks* Budapest, Proceedings of 3rd Int.Workshop on Design of Reliable Communication Networks pages:236-242

[10] B. Leuf. Peer to Peer: Collaboration and Sharing over the Internet Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA 2002

[11] S. Saroiu, P. K. Gummadi, S. D. Gribble. A Measurement Study of Peer-to-Peer File Sharing Systems, Department of Computer Science & Engineering, University of Washington. *Proceedings of the Multimedia Computing and Networking (MMCN)*, San Jose, January, 2002.

[12] J. Lygeros. Hierarchical, Hybrid Control of Large Scale Systems. *California PATH Research Report* UCB-ITS-PRR-96-23

[13] J. Liu, X. Liu, T. J. Koo, B. Sinopoli, S. Sastry, and E. A. Lee. A Hierarchical Hybrid System Model and Its Simulation, *Proceedings of the 38' Conference on Decision & Control*, Phoenix, Arizona USA December 1999

[14] W. Hu, N. Bulusu, and S. Jha. A Communication Paradigm for Hybrid Sensor/Actuator Networks, *International Journal of Wireless Information Networks*, Vol. 12, No. 1, January 2005

[15] Sacha Krakowiak, Middleware Architecture with Patterns and Frameworks, *Distributed under a Creative Commons license http://creativecommons.org/licenses/by-nc-nd/3.0/* February 27, 2009

[16] P. A. Bernstein. Middleware An Architecture for Distributed System Services, Digital Equipment Corporation Cambridge Research Lab 1993

[17] M. Chouiten, J. Didier, M. Mallem. Component-based middleware for distributed augmented reality applications, *5th International Conference on Communication System Software and Middleware* (COMSWARE '11), Verona : Italy (2011)" DOI : 10.1145/2016551.2016554

[18] A . M. Khandker, P. Honeyman, T. J. Teorey. Performance of DCE RPC, Center For Information Technology Integration, The University of Michigan, Services in Distributed and Networked Environments, 1995.

[19] K. Arnold, R. Scheifler, J. Waldo, B. O'Sullivan, A. Wollrath. Jini Specification, 1st Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA ©1999, ISBN:0201616343

[20] J. Waldo. The Jini architecture for network-centric computing. Sun Microsystems, Burlington, MA, Published in: · Magazine Communications of the ACM CACM Homepage archive Volume 42 Issue 7, July 1999

[21] P. T. Eugster, P. A. Felber, R. Guerraoui, A. Kermarrec. The many faces of publish/subscribe, ACM Computing Surveys (CSUR) Volume 35 Issue 2, June 2003 Pages 114 – 131

[22] E. Souto, G. Guimaraes,G. Vasconcelos, M. Vieira, Nelson Rosa, C. Ferraz, J. Kelner. Mires: a publish/subscribe middleware for sensor networks, Pers Ubiquit Comput (2006) 10: 37–44 DOI 10.1007/s00779-005-0038-3

[23] L. Opyrchal and A. Prakash. Publish Subscribe Middleware, The University of Michigan USA

[24] H.Penny Nii. Blackboard Systems, Knowledge Systems Laboratory, Standford University

[25] D. D. Corkill. Blackboard Systems, Blackboard Technology Group, Inc.

[26] Honkola, Jukka.Smart-M3 information sharing platform. Nokia Research Center, Helsinki, Finland, *Computers and Communications (ISCC), 2010 IEEE Symposium* on, June 2010

[27] Nakevska, M., Vos, C., Juarez Cordova, A.G., Hu, J., Langereis, G.R. & Rauterberg, G.W.M. (2011). Using game engines in mixed reality installations. In Anacleto, J., Fels, S., Graham, N. & et al. (Eds.), Entertainment Computing – ICEC 2011, (pp. 456-459). Berlin: Springer.

[28] Nakevska, M. Juarez, A., Hu.J. CryVE: Modding the CryEngine2 to create a CAVE system

[29] M. Mizuno, M. Raynal, J. Z. Zhou. Sequential Consistency in Distributed Systems. Dept. of Computing and Info, Sciences, Kansas State University, Manhattan

[30] J. Hu. Design of a Distributed Architecture for Enriching Media Experience in Home Theaters, in Department of Industrial Design, Eindhoven University of Technology: Eindhoven, 2006.

[31] J. Coutaz,. PAC, an Implemention Model for Dialog Design, in Interact'87, 1987, Stuttgart.

[32] J. Coutaz. , PAC-ing the Architecture of Your User Interface, in 4th Eurographics Workshop on Design, Specification and Verification of Interactive Systems: Springer-Verlag, 1997.