Context-Based Generation of Multimodal Feedbacks for Natural Interaction in Smart Environments

Didier Perroud, Leonardo Angelini, Omar Abou Khaled, Elena Mugellini
Department of Information and Communication Technology
University of Applied Sciences of Western Switzerland
Fribourg, Switzerland
{didier.perroud, leonardo.angelini, omar.aboukhaled, elena.mugellini}@hefr.ch

Abstract— **This paper presents a complete system for the generation of multimodal feedbacks in smart environments. It is based on an underlying framework, NAIF, which allows taking into account context information in order to choose the better outputs for the user, improving the effectiveness of feedbacks and the naturalness of the interaction. The extension of NAIF is composed by three main components: a fission engine, a context engine and a context service. Several examples are provided to show the working principles of the developed system. The extended NAIF framework allows setting up applications for the delivery of consistent and quality multimodal feedbacks in heterogeneous environments.**

Keywords-Multimodal feedbacks; context-aware; smart environment; reasoning engine; multimodal fission.

I. INTRODUCTION

Technology is spreading in our homes, many objects become smarter and many devices, such TVs or media-players, have now Internet access. However, most of them offer functionalities that have a limited scope; they often neither communicate with other near devices, nor try to retrieve information from the environment to modify their behavior. Nowadays, many devices can communicate in our homes through protocols like DLNA (Digital Living Network Alliance) [14]; however, the functionality of these protocols is limited to the share of media content. Frequently, the heterogeneity of household devices forces the user to adapt to each system, often without the possibility to choose the best way to communicate with them, for example using different modalities.

Since 1991, Weiser stated the importance of having technologies that disappear to the human sight, integrated in our lives as imperceptible companions [11]. Computers are now smaller, easy to bring with us or embedded in small objects or either in textiles; thus, actually, they are physically disappearing. However, these devices often neither take into account the user perspective, nor they exploit the information from the environment to ameliorate the user perception. In order to weave all these devices together and to create a smart environment, it is necessary to allow them to communicate, to obtain context information and to share embedded resources. These resources can be computational capabilities, information from sensors, stored data, audio and video outputs and actuators.

In this paper, we present a system for the generation of multimodal feedbacks, which takes advantage of the heterogeneous set of devices that can be found in a smart environment. These devices, thanks to their embedded sensors, can contribute also to the generation of context information, which is used to provide consistent and efficient feedbacks to the user. The proposed system is based on the Network Ambient Intelligent Framework (NAIF), a framework for smart environments that has been previously presented in [5]. We have extended NAIF introducing an entity-based context model, a reasoning engine based on first order logical trees and the management of feedbacks using context information. Furthermore the system is able to combine multiple outputs according to the model proposed by Vernier and Nigay [8]. This framework is intended to be easily extensible with new information about context and new heterogeneous devices that can offer input or output resources. Several applications can use these resources at the same time to deliver messages to the user taking advantage of different modalities.

The paper is structured as follows: Section 2 reviews existing projects that are most related to our work. The third section presents the architecture and the implementation details of the system. Finally, we provide three practical examples based on the proposed system, which illustrate its potentialities in different scenarios. The three examples aim to deliver a multi-content message from the well-known social network Twitter, using context information to assure that the multimodal feedback has the best suitability for the conditions of the environment. Future improvements and conclusion end the paper.

II. RELATED WORK

According to the paper of Dumas et al. [1], the use of natural means of communication facilitates the human interaction with the machine. Communicating using different interaction modalities can be definitely considered as an innate characteristic of human beings. This is generally addressed in the machine through the fusion of inputs modalities and the fission of outputs. Foster defined the multimodal fission as “the process of realizing an abstract message through output on some combination of the available channels” [3]. In our work, we perform multimodal fission in an intelligent environment, which claims to be non-intrusive for the human. Multimodal fission is a research topic that is not often addressed in the scientific community. Although multimodality is a popular topic of research, papers on multimodality address most frequently the fusion process that aims to interpret signals as input. Most systems that use multimodality consist in a natural dialog application, like the European project SmartKom [9], which aims to

 ISBN: 978-1-61208-235-6

design a virtual assistant that communicates with the user through gestures and voice. In the SmartKom project, multimodality is limited to a few specific modalities and the fission process cannot be applied to a distributed smart environment. Foster presented in 2003 a survey of multimodal fission [3]. This document is a deliverable of the project COMIC, a dialogue system that uses multimodality. The output generation of the COMIC project is described in [2]. Although the state of the art proposed by Foster is several years old, some key points of the fission process are still very relevant. Foster explains that the fission process can be separated into three parts. The first part is the content selection and the structuring of the information. The second part is the selection of modalities based on context knowledge. The final part deals with the coordination of the outputs in order to provide consistent feedbacks. Our system is not focused on the content selection, but implements a selection of modalities based on the context retrieved from an intelligent environment.

Vernier and Nigay presented in [8] a conceptual space about the output combination of modalities. The authors explain that multimodal outputs can be generated according to five different compositions. These five compositions can be further interpreted according to different aspects: time, space, articulation, syntax and semantics. Our work is based on this conceptual space to perform the task of output coordination in the fission process. The fission process is performed as part of an intelligent environment; we propose a composition of output modalities in time and space and the control of generators according to the semantic aspect of composition. The possible compositions of these aspects are presented in Section 3.

The work presented in this article is an extension of the previous approach that we proposed in [5]. This approach was intended to propose a solution for the generation of context-aware multimodal feedbacks based on NAIF, which aims to set up and operate a smart environment. The main components introduced in that article were a context service, a reasoning engine for context and a fission engine. Although a simple working prototype has been already presented, many parts of this concept were neither implemented, nor detailed. Indeed, the fission process and the reasoning techniques were not defined and no context modeling was proposed. Our contribution aims to overcome these different lacks. We propose a context model and a reasoning engine that aim to exploit the context of a smart environment. We also present a multimodal fission process based on our context management solution.

The multimodal fission process proposed in this paper is based on the context of the environment. One of the main challenges of exploiting context information is to model it. The scientific community is very active on this topic. In the literature we find generally three mains trends about context modeling. The first trend is ontology-based. Wang et al. recommend the use of this technique in [10]. Ontologies also offer the ability to use inference to reason on context. However, Henricksen et al. use an object-oriented model in their work [4]. This second technique facilitates the implementation of context management, thanks to its similarity with objects-oriented programming languages. Finally, Ranganathan et al. proposed a modeling technique based on predicates [6]. This technique allows easily applying advanced mathematical reasoning on context information. Each mentioned modeling technique has its strengths and weaknesses. As shown by Ranganathan et al. in [7], an interesting approach is to use several techniques simultaneously. Indeed, they used an ontology model to share vocabulary and a predicate model for the implementation. A Survey of context modeling and reasoning techniques can be found in [16]. In our work, we have chosen to use an object-oriented model to facilitate the implementation of context management. In fact, the context management solution is integrated into a Framework that must be used by several developers that have their own representation of the context.

III. System Architecture

NAIF operates in three layers: the inter-communication

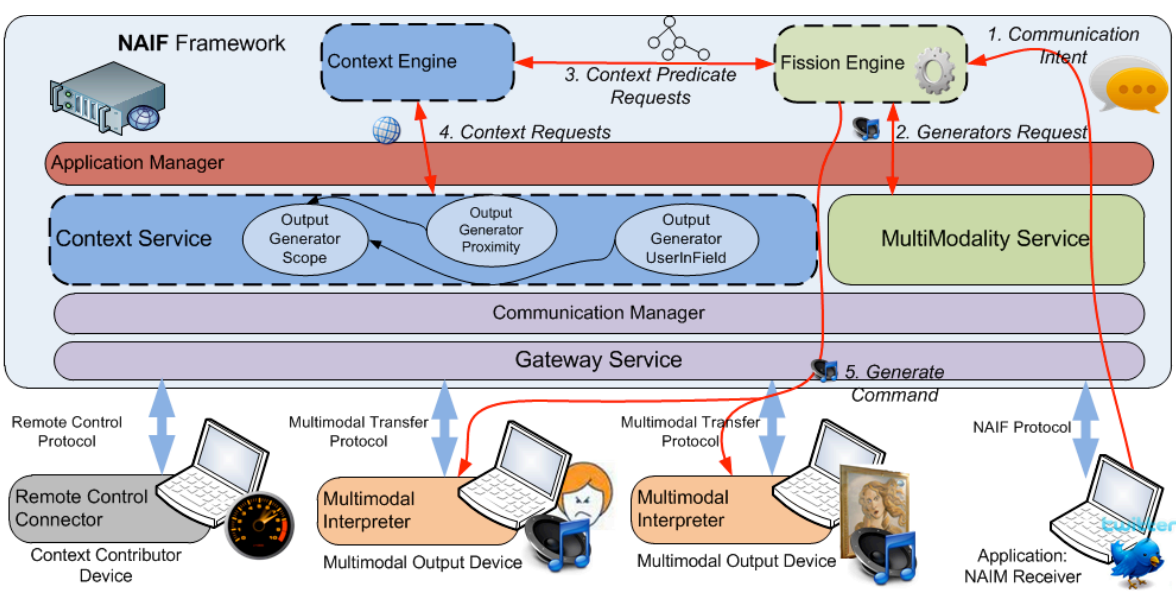

Figure 1. NAIF Extended framework for context-aware multimodal feedback; dashed lines show improved components.

 ISBN: 978-1-61208-235-6

layer, which includes the gateway and the communication manager, an interworking layer, which offers services and shared resources to the applications, and the application layer, which can handle applications running on the devices connected to NAIF. Further details about the architecture could be found in [5]. In order to generate multimodal feedbacks suitable for the context of the environment, NAIF has been improved in several parts, depicted in Figure 1. First, the Context Service for context generation has been ameliorated with an object-oriented entity model for context representation. Second, the Context Engine has been extended to handle requests for context retrieval, using first order logical trees. Finally the Fission Engine has been improved, taking advantage of the new Context Engine, and introducing spatial, temporal and semantic compositions of the feedbacks. NAIF is extensible to several applications and heterogeneous devices that want to communicate information to the user or offer proper data for the generation of the context. These devices can benefit from outputs available in the environment, publish sensor information and output capabilities according to the three respective protocols depicted in Figure 1: the NAIF Protocol with communication intents, the Remote Control Protocol and the Multimodal Control Protocol. Both the Fission Engine and the Context Engine run in the application layer, thus they can be easily modified or replaced without affecting the operation of the whole framework. In the following paragraphs (A, B and C), the principles of operation of the main components for context-based generation of multimodal feedbacks are explained. The five steps necessaries to the generation are shown in Figure 1.

An application that wants to produce an output has to send a *communication intent*, a structured message that specifies to the Fission Engine the type of media, an eventual recipient and the composition of information, if more media are present (Step 1). Multiple media are handled according to three of the five composition aspects presented by Vernier and Nigay [8]. An adaptation of their model for our system is shown in Figure 2. Communication intents can specify temporal composition parameters, i.e., the delay between each message, and a spatial composition, for example if two media should be played in adjacent or opposite displays. Finally, the semantic attribute specifies if the media that are present in a communication intent are semantically complementary, thus if both must be reproduced in order to correctly deliver the information. When a communication intent is received, the Fission Engine retrieves the list of the available Output Generators from the MultiModality Service (Step 2). Generators are filtered according to their capabilities of displaying media, to the scope or recipient of the message, spatial composition and context requirements. Each time that context information is needed, a logical tree request is formulated by the Fission Engine and sent to the Context Engine (Step 3), which retrieves data from the Context Service (Step 4). Finally, generation commands are sent to the chosen outputs, which, if necessary, will be locked to prevent further uses by other applications while displaying the feedback (Step 5). Temporal composition attributes are used to synchronize outputs, if specified. Further details are presented in the following paragraphs.

A. Context Model and Generation

The context is represented by an object-oriented model and is maintained up-to-date by the Context Service. Each Entity is defined in a stand-alone XML Schema that contributes to the global model. The choice of an object-oriented approach instead of an ontology model is made to ensure rapid development and easy extension even for novice programmers. The model is based on Context Entities and a Context Entity Manager for each entity type. The Context Entity Manager generates and updates the corresponding entities following three different strategies. These strategies are implemented according to the context acquisition methods proposed by Mostefaoui et al. in [17]. For Sensed Context Entities, e.g., Luminosity or Noise, it retrieves necessary data from the NAIF Remote Control Service, which collects measured data from sensors embedded in the environment or in the connected devices. For Knowledge Based Context Entities the Context Entity Manager obtains data from local (e.g., sensors positions) or online (e.g., weather) databases. The last type of Context Entity Manager handles Derived Context Entities combining

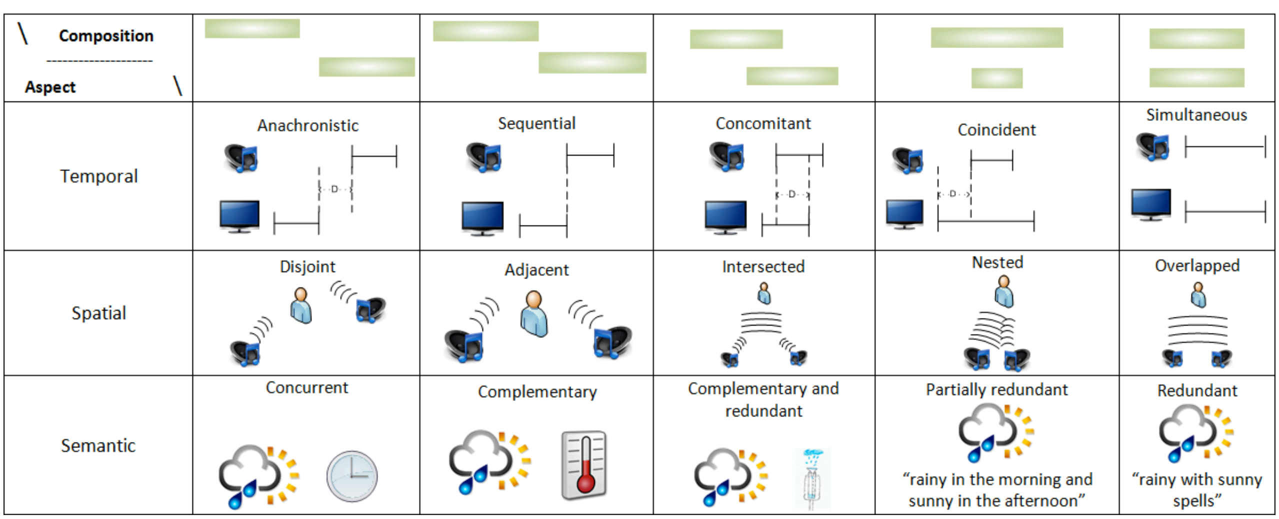

Figure 2. Spatial, temporal and semantic composition of outputs according to the model of Vernier and Nigay.

 ISBN: 978-1-61208-235-6

and elaborating information from one or more entities. Adding a new Context Entity to the global model is as easy as defining it in an XML Schema and implementing the Context Entity Manager, which retrieves data from sensors and transform them to high level thresholds. These thresholds, e.g., high, low or medium, are easier to understand for an end-user and they are fixed by the programmer during the system configuration. For the purpose of the adaptive multimodal feedback generation, we have defined a specific context model that describes the state of the Output Generators. This basic model is showed in Figure 1 in the Context Service module. The first type of entity, Output Generator Scope, describes a generator and specifies the type, the range or scope, and the position. These entities belong to the Knowledge Based Context Entity type and will not be automatically created by the NAIF Framework, but generally must be defined by the programmer. The Output Generator Proximity Entity aggregates information of two Output Generators and determines if they are reciprocally near or far. Similarly, the Output Generator UserInField, determines if a person is within the scope of an Output Generator or not. Only available Output Generators are present in the context state. The Person Entity used for the tests contains just the name and the ID of the associated RFID tag. The model obviously involves also other entities like Luminosity, Noise, etc., which will not be described in detail.

B. *Context Engine*

The Context Engine allows handling complex queries on the context state, using a protocol formalized by first order logical trees. Both requests and results are formalized by a tree of entities from the context state, shown in Figure 3. A first order logical tree is a logical expression (AND/OR), which involves entities values, or recursively, other sub-trees. Each leaf of the tree is represented by an entity state. In the request, desired entities states are specified; to elaborate the query result, the correspondent entities are retrieved with the desired states from the Context Service: if there are no corresponding entities with the specified state, the leaf has a false logical value; otherwise the corresponding entities are put in the result tree and the logical value is true. Providing a tree of entities as a result is very important to retrieve information about selected entities. Moreover, each branch of a tree can be marked as *required* or as *optional*. If a branch is optional, it will not affect the truthfulness of the parent expression, however the corresponding entities, if their expressions are true, will be included in the query result. Optional branch of predicates are also useful to

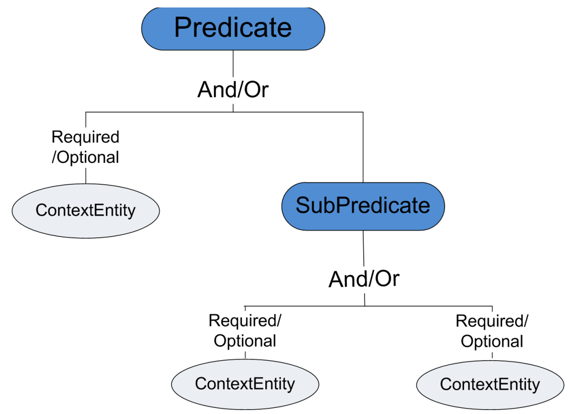

Figure 3. First-order logical trees for context requests.

retrieve information that can be used to ameliorate the quality of feedbacks when basic constraints are verified. An example of context request is presented in Figure 4. An application wants to send a video message to Bob and Alice. The Fission Engine build the request asking desired context states for four different entities, but only two of them are marked as required. The example supposes that in the Context Service the Noise state is low, the Luminosity is high and that Bob and Alice are in proximity of the TV. It is worth noting that the Luminosity entity state is not included in the response because it is marked as optional and has a false logical value. The Context Engine can accept request also from other applications that want to directly access context information without delegating the work to the Fission Engine.

C. *Fission Engine for Multiple Feedbacks*

The role of the Fission Engine is to choose a proper output and modality for high-level media, e.g., text that can be displayed to the user either as visual feedback or as aural feedback, using Text To Speech (TTS). The Fission Engine makes this choice according to the context information and an eventual spatial composition. The overall selection process has been already described in Figure 1. In this paragraph, the selection process of the Output Generators and their management during the generation of feedbacks is

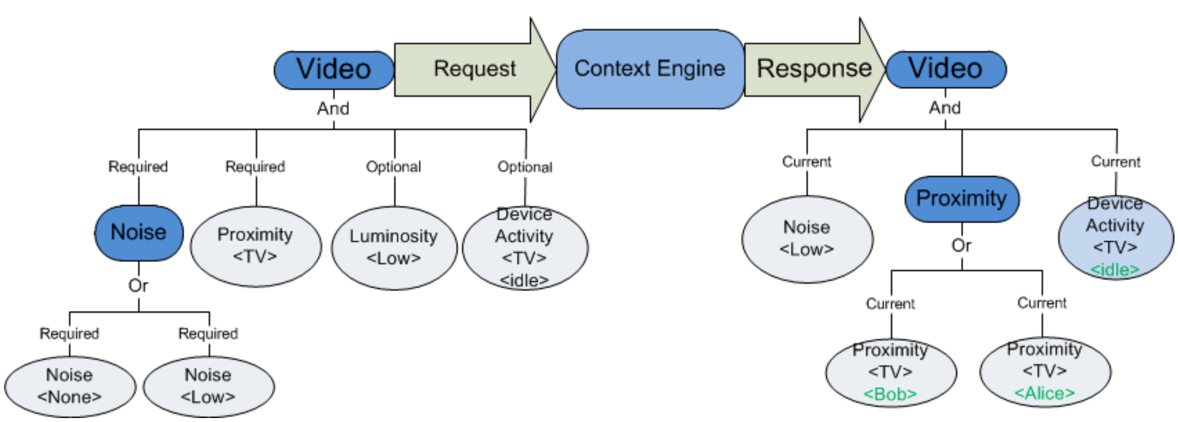

Figure 4. Example of context request.

detailed.

Figure 5 resumes the workflow of the Fission Engine and shows the related exit points, which occur when one or more requirements are not satisfied and the generation of the outputs is not possible. In few cases, detailed in this paragraph, the generation is not stopped even if some constraints or requirements are not fulfilled. First of all, the Fission Engine checks if the types of media required by the communication intent are in the list of compatible types (for example Text, Images, Emotions, etc.). Then, the available generators are retrieved from the Multimodal Service and filtered according to the media types of the communication intent. Output Generators are further filtered according to their scope: for messages without a recipient, global generators (available for all people in the room) are chosen. Otherwise, if a recipient is specified, the Fission Engine formulates a request to the Context Engine in order to determine which Output Generators are in the same scope of the recipient. The current implementation can handle only a single recipient. Applications that want to send messages to multiple users should replicate the communication intent for each recipient. For global scope multiple feedbacks, spatial combination is taken into account and the available spatial combinations are then filtered according to the context requirements. The spatial combination for multiple feedbacks is chosen among all possible permutations of Output Generators. This technique is acceptable for small environments but becomes very inefficient when many Output Generators are present and should be improved for this latter case. The choice of the better spatial combination (or of the single Output Generator for communications composed by only one media) is made according to context requirements for each type of modality for that media. Requirements are generally defined by the programmer in a database and are used to construct the context requests as defined in the previous paragraph. When there are no generators with an adequate context, the behavior of the Fission Engine changes according whether the communication intent has more than one media or only one.

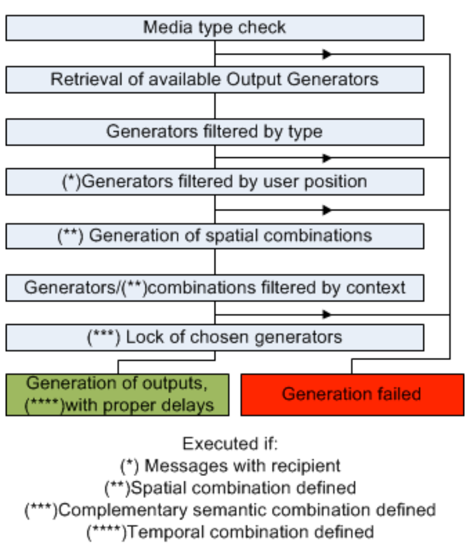

Figure 5. Fission engine workflow.

In the first case, the Fission Engine sends the output commands to the chosen generators even if the context does not respect the defined requirements, because the multiple nature of the communication can probably help the user to understand the message even if some outputs are perturbed (Assumption 1). If only one media is present, instead, we designed the Fission Engine to make the communication fail when none of the generators has an adequate context. Thus, we believe that using only one perturbed generator it is not possible to correctly deliver the information to the user (Assumption 2). The Assumption 1 is probably correct only if the semantic composition of the message is *redundant*. In the case of *complementary* information, the Fission Engine should drop the communication intent. However, in the current implementation of the system, the Fission Engine is not able to choose generators according to *semantic combination* of the information, nor to assign more generators to the same piece of information, in order to augment the quality of the feedback in the case that all the available generators are perturbed. However, semantic information is taken into account during the generation of feedbacks. In fact, Output Generators that display non-redundant information are locked by the Fission Engine in order to avoid that other applications could interrupt the output generation. Delays are applied to the generation of multiple messages in order to respect the desired temporal combinations.

IV. SCENARIO

The main objective of the proposed system is allowing a seamless interaction in our homes using existing technologies, just interconnecting current available devices through the NAIF Framework. We developed an application, Natural Ambient Internet Messaging (NAIM), which uses the Context Engine and the Fission Engine of NAIF to deliver proper feedbacks to the users in a smart environment. NAIM is a simple connector to the popular social network Twitter. For our purpose, i.e., testing the multimodal fission of NAIF, we developed only a receiver that is able to interpret messages written on a dedicated account with a proper syntax. The syntax allows defining in a tweet several messages at one time, which can involve different type of media: text, emotions and images. Moreover, the composition of these messages can be specified according to the spatial, temporal and semantic axes. This is done by adding special text commands at the end of the tweet as shown in Figure 6.

Figure 6. A local scope message (green, in red the recipient) with emoticon (orange) and a global message (green) with spatial and temporal composition (yellow).

In the following paragraphs, we analyze three different scenarios based on the NAIM Receiver that show how the

Figure 7. Scenario: 1) TV; 2) Beamer; 3) Aphrodite; 4) Laptop.

multimodal context-aware generation can benefit to the Human Computer Interaction within a smart environment. The three scenarios have been implemented and tested in a smart living room where several devices were connected through NAIF. Each device was connected through an Ethernet or Wi-Fi connection to another PC that was running the NAIF base Framework (Gateway), the Context Engine, the Fission Engine and the NAIM Receiver. Obviously, these three latter components are applications that could run on any computer connected to the NAIF Gateway. In particular, as shown in Figure 7, we used a media center connected to a TV, a Personal Computer connected to a beamer and a laptop for local scope messages. Context information is obtained using popular Phidgets [15], in particular a microphone and a luminosity sensor. An RFID reader has been used to detect the presence of a target user.

Once a tweet is received by the NAIM Receiver, the time necessary to generate feedbacks was within the order of magnitude of the transmission time of the messages over the local network. The major delay between an input and output of this demonstrator is related to the time necessary to Twitter servers to store and send back the tweet to the NAIM Receiver through their services.

The NAIM Receiver decomposes the tweet and builds a communication intent composed of more than one media. Three different media have been implemented in this scenario: text, images and emotions. For text messages, two output modalities are available: a visual one, which displays the text in a window, and an aural one, which uses Text To Speech (TTS). Obviously, images can be displayed only with a visual feedback. Emotions, instead, are represented either with visual smiley or by animating a physical painting called Aphrodite [12]. Aphrodite is able to represent emotions through sounds and expressions of the face in the painting. The first media of a communication intent is always the text "New message received by SenderName", where SenderName is the Twitter account that published the message.

A. *Global Text Message in a Noisy Environment*

The first scenario involves a global message to be displayed to all the users, e.g., an alert message. This message, which is sent as text media, will be displayed as an aural feedback using TTS if the noise of the environment is low or as a visual feedback on available screens otherwise. When the tweet is received by the NAIM Sender, it generates a communication intent that includes two text media: the standard text with the sender of the tweet and the content of the tweet. Then, the Fission Engine requests global scope Output Generators that are able to display text, i.e., in our scenario, the TV and the beamer. For each generator, two modalities are available to display text: an audio modality (TTS) and a visual modality (text displayed in a window). In this scenario no constraint has been set for the luminosity; the TV audio generation (TTS), instead, had a constraint on the noise measured by the associated microphone. This constraint has not been set for the beamer, which was paired with more powerful speakers. Further, the Fission Engine controls if the constraint is verified and chooses the best output generator for each of the two media. When only the TV is turned on and there is noise in the room, both messages will be displayed with visual feedbacks; otherwise, with no noise, the system generally prefers to use both visual and aural feedbacks, assigning one of this modality to each of the two messages. If also the beamer is turned on, the Fission Engine will choose the TV for the text display and the beamer for TTS. Introducing a new constraint on the context of the room in order to provide visual feedbacks on the beamer only if the luminosity is low will lead the Fission Engine to use TTS on the beamer or to display both messages as text on the television. This example shows how the system can automatically improve the quality and the efficacy of the feedbacks.

B. *Messages Addressed to a Mobile User*

The same scenario is now considered with messages with a specified recipient. The user is authenticated using an RFID tag that is recognized by a computer with a local scope Output Generator. The target application would like to provide feedbacks to the user even when he or she is moving in the home, choosing each time a local Output Generator and the best output modality according to noise or luminosity information. The Fission Engine in this case analyzes the recipient field of the communication intent and finds proper local generators according to the information of the Output Generator UserInField entity.

C. *Multimedia Message with Context-adaption*

Previous examples involve media of only one type: text. In order to demonstrate the multimodal fission of a message, in this scenario emotions (emoticons) have been included. While text will be displayed either with visual or aural feedbacks, emotions will take advantage also of the aforementioned Aphrodite system, which is able to display them changing facial expressions in the painting. However, this picture cannot be seen with low luminosity: in this latter case the system will prefer to display the emotion as a smiley on a screen, for example the beamer. On the other hand, a smiley could not be displayed on the beamer if the luminosity of the room is high. Similarly to the first scenario, it is possible to define these context requirements in order to allow the Fission Engine to choose the best Output Generators.

V. Conclusion and Future Improvements

This paper presented a system that introduces an object-oriented model to represent context, a Context Service to maintain up-to-date it, a Context Engine to handle requests on the context state and verify desired conditions and a Fission Engine to select the best outputs for a given communication intent. An application that delivers messages from a Twitter account to the user has been developed in order to demonstrate the working principle of the system. NAIF with the management of multimodal feedbacks according to context information allows the delivery of high quality feedbacks within a smart environment, taking advantage of input and output resources available on present devices. The system allows to several applications to share these resources dynamically, building up a model of the context and choosing proper resources according to context information.

Obviously some limitations are present: NAIF lacks a system of coordinates that could allow an accurate definition of the position of the Output Generators, of the user and of the context sensors. By now, the positions of the generators are defined by giving a name to the zones where they are located, with no definition of proximity and relative distance. A system of coordinates and the tracking of users as that shown in [13] should be included in order to enhance the capabilities of NAIF and the quality of the feedbacks with spatial composition. Moreover, types of media, context requirements and Output Generators capabilities are defined with XML files, which are not very intuitive to be configured by end-users. A graphical user interface should be implemented in order to allow the user to easily set up a new system based on NAIF.

References

[1] B. Dumas, D. Lalanne, and S. Oviatt, “Multimodal interfaces: a survey of principles, models and frameworks”, Human Machine Interaction, 2009, pp. 3-26,

[2] M. E. Foster, “Producing multimodal output in the COMIC dialogue system”, Proc. of the W3C Workshop on Multimodal Interaction, 2004

[3] M. E. Foster, “State of the art review: Multimodal fission”, COMIC project Deliverable 6.1, 2002 [retrieved: July, 2012]

[4] K. Henricksen, J. Indulska, and A. Rokotonirainy, “Modeling context information in pervasive computing systems”, Proc. of Pervasive, 2002, pp. 79-114.

[5] D. Perroud, L. Angelini, E. Mugellini, and O. A. Khaled “Context-aware multimodal feedback in a smart environment”, AMBIENT 2011, pp. 1-6.B.

[6] A. Ranganathan, J. Al-Muhtadiand, R. H. Campbell, “Reasoning about uncertain contexts in pervasive computing environments”, IEEE Pervasive Computing, 2004, pp. 62-70.

[7] A. Ranganathan, R. E. McGrath, R.H. Campbell, and M. D. Mickunas, “Ontologies in a pervasive computing environment”, Proc. of the Workshop on Ontologies and Distributed Systems, 2003,

[8] F. Vernier and L. Nigay, “Multimodal interfaces: composition and characterizazion of output modalities” (Interfaces multimodales: composition et caractérisation des modalités de sortie), Proc. of ErgoIHM'2000, 2000, pp. 203-210.

[9] W. Wahlster, “SmartKom: fusion and fission of speech, gestures and facial expressions”, Proc. of the 1st Int. Workshop on Man-Machine Symbiotic Systems, 2002, pp.213-225.

[10] X. H. Wang, D. Q. Zhang, T. Gu, and H. K. Pung, “Ontology based context modeling and reasoning using OWL”, Proc. of the 2nd IEEE Pervasive Computing and Communications, 2004, pp. 18-22.

[11] M. Weiser, “The computer for the 21st century”. Scientific American, 1991, 265 (3), pp. 94-104.

[12] Aphrodite, a head capable of reproducing one dozen of human expressions http://www.aphrodite-glance.ch, [retrieved: July, 2012]

[13] Mori, T., Suemasu, Y., and Noguchi, H., “Multiple people tracking by integrating distributed floor pressure sensors and RFID system”, *Systems, Man and Cybernetics*, (2004), pp. 5271-5278.

[14] Digital Living Network Alliance, “DLNA Home Networked Device, Interoperability Guidelines v1.0,” Jun. 2004 [retrieved: July, 2012]

[15] Phidgets, Products for USB Sensing and Control, http://www.phidgets.com/ [retrieved: July, 2012]

[16] C. Bettini, O. Brdiczka, K. Henricksen, J. Indulska, D. Nicklas, A. Ranganathan and D. Riboni, “A survey of context modelling and reasoning techniques.” Pervasive and Mobile Computing 6, 2010, pp.161-180.

[17] G.K. Mostefaoui, J. Pasquier-Rocha, and P. Brezillon, “Context-Aware Computing: A Guide for the Pervasive Computing Community”, Proc. of the IEEE/ACS International Conference on Pervasive Services, 2004. pp 39-48.