Designing Patterns for Multimodal Fusion

Ahmad Wehbi, Amar Ramdane-Cherif, Chakib Tadj

Abstract. *This paper presents a design of an architecture that facilitates the work of a fusion engine. The logical combination or merging of input streams invoked by the fusion engine is based upon the definition of a set of patterns and its similarity with previously collected data from various modalities. Previous fusion engine designs had many weaknesses, among them their being specialized on a specific domain of application. The proposed architecture addresses such weakness and provides additional features, namely its ability to handle large number of modalities and due to its using knowledge base and standardization characteristics; it becomes suitable to various types of multimodal systems. The techniques used to achieve these features are discussed in this paper.*

Key words: *modality, patterns, XML, data interpretation, knowledge base, normalization.*

1 Introduction

As per W3C (http://www.w3.org/2002/mmi/) specification, a multimodal interaction activity seeks to allow users to dynamically select the most appropriate mode of interaction for their current needs, including consideration of any disability whilst enabling developers to provide an effective user interface for whichever modes the user selects. Depending upon the device, users will be able to provide input via speech, handwriting, and keystrokes, with output presented via displays, pre-recorded and synthetic speech, audio, and tactile mechanisms such as mobile phone vibrators and Braille strips. Various multimodal applications [1] are conceived and are effective solutions for users who have constraints such as the impossibility of using a keyboard or a mouse [2], having visual handicap [3], being mobile using wireless telephone/mobile devices [4], and being weak and disabled [5]. These applications are integrated with web services which are application functions or services. These services can be accessible from another application (a client, a server or other web services) within the Internet network using the available transport protocols [6]. Application service can be implemented as an autonomous application or as a set of applications. It pertains to a technology allowing applications to communicate with one another via Internet, independently of the operating system platforms and languages that these applications are based. In the literature, several solutions were proposed to facilitate the work of fusion engines. To our point of view, the most significant works follow. Engel et al [7] proposed an approach for processing modalities in a system called *SmartKom* [8]. Its basic idea is to generate all meaningful combinations after considering all hypotheses and afterwards selects the *n* best results which are passed to the intention analyzer. Apart from its relatively easy implementation, its other advantage is that during the processing, no decisions based on incomplete context have to be made. The disadvantage of this approach is that under adverse circumstances, specifically in the case of recognition errors or unintended gestures performed by the user, the generation of all meaningful combinations takes too much time. Sonntag et al [9] proposed an ontological solution for a system called *SmartWeb*. This system is based on question answering technology that combines different kinds of domain ontologies into an integrated and modular knowledge base.

CompSysTech'11, June 16–17, 2011, Vienna, Austria.
Copyright ©2011 ACM 978-1-4503-0917-2/11/06...$10.00.

For this purpose, they defined an upper model ontology based on SUMO (www.ontologyportal.org) and DOLCE (www.loa-cnr.it/DOLCE.html) and integrated each domain ontology on it. The main problem of this approach is the specification of its design. Since it is only for the application of answering questions, the solution presented by this work is very limited. The architecture of*HephaisTK* system developed by Dumas et al [10] is based on software agents that are dispatched to manage individual modality recognizers, receive and encapsulate data from the recognizers, and send them to an individual central agent named the "*postman*". This postman agent centralizes all data coming from the dispatched recognizers agents in the database, and distributes the data to other interested agents like the fusion manager. However, this architecture needs a configuration file to be specified for describing the human-machine multimodal dialog desired for the client application, and for the specification to which recognizers need to be used. Having taken these weaknesses into account, we come up with a proposed architecture that addresses these issues. This is done through modeling patterns that deal with different modalities, and by creating a knowledge base that contains these patterns. The adoption of this architecture will facilitate the work of a fusion engine by giving it the most meaningful combinations of data. The paper is organized as follows. In section 2, we present the architecture itself and describe its components. In section 3, we present a scenario demonstrating how the architecture works and finally we conclude this paper in section 4.

2 Patterns Architectural Design

In this section, we will describe our proposed architectural design with focus on the use of patterns as a solution to the described problems of previous systems architectures

2.1 Overview Of The Pattern's Architectural Design

The previous diagram in Fgure 1 is further refined in Figure 2 showing the whole concept of the architecture. Aside from the recognition, interpretation, and fusion processes that were described earlier, the refined architectural design shows new components, namely the knowledge base and the database. The remaining system components and the description of various steps involved in the fusion process are described below:

Knowledge Base: It is a container that stores the patterns as ontology concepts.

Database: It is a container that stores all probable meaningful combinations of interpreted data. These combinations are used by the fusion engines by merging them to obtain a final result.

Step 1 – User - Modality: In this step, the user begins his interaction with the system, using various modalities such as voice, body movement, eye gaze, facial expression, etc. By performing an action, the user automatically launches/activates modalities.

Step 2 – Modality – Recognition of Modalities: This module receives as input XML files containing data concerning modalities, usually captured using sensors (e.g. webcam, micro, touch screen, etc.). From each XML file, this module extracts some tag data that it needs for fusion. Afterwards, it creates a resulting XML containing the selected modalities and each modality's corresponding parameters. In conformity with W3C standard on XML tags for multimodal applications, we use EMMA. EMMA [11] is a generic tagging language for multimodal annotation. It is an integral part of the W3C norm for multimodal interactions. Its general objective is to automatically represent the information extracted from user inputs through interpretation of system components. The EMMA tags represent the semantically recovered input data (e.g. gests, speech, etc.) that are meant to be integrated to a multimodal application. EMMA was developed to allow annotation of data generated by heterogeneous input media. When applied on target data, EMMA result yields a collection of multimedia, multimodal and multi-platform information as well as all other information from other heterogeneous systems.

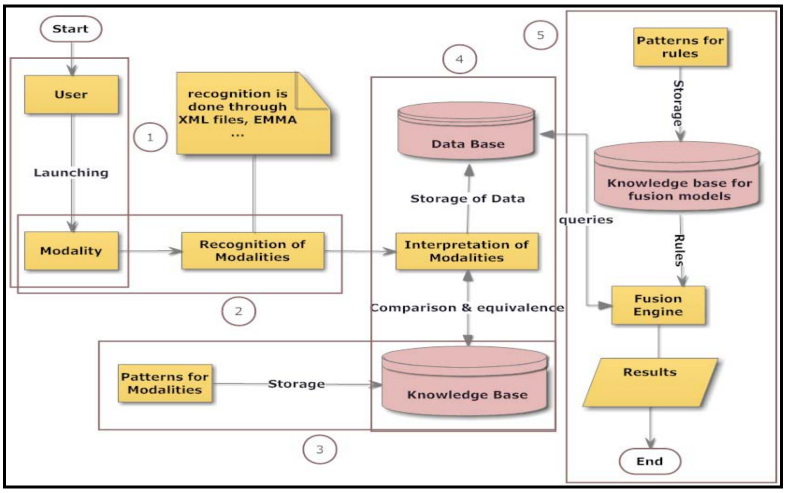

Figure 1: Architecture overview

Step 3 – Patterns – Knowledge base: Patterns are stored as concepts in the knowledge base which contains data format, parameters and modality events. These patterns are used by the interpretation module to check a similarity match between the data provided by modalities against the patterns themselves. These concepts are semantically represented inside the knowledge base using the *Web Ontology Language* (OWL). OWL (http://www.w3.org/TR/owl-features/) is a vocabulary extension of Resource Description Framework or RDF (http://www.w3.org/TR/owl-ref/) semantics. Thus, any RDF graph forms an OWL full ontology. Furthermore, the meaning given to an RDF graph by OWL also includes the meaning given to the graph by RDF.

Step 4 – Interpretation of Modalities: This module is responsible for checking if there is a match between the data associated with modalities and that of the patterns. Match checking is done by determining the following conditions: (1) the *data format* - the data format from a special modality is checked with the data format of the pattern that concerns that modality, (2) *parameters* – these are also checked to verify the most suitable selection of modalities. This verification is done in consideration of the user's context, the environmental context and the system context, and (3) the events – this is done by comparing the present event, meaning the action done by the user concerning the specified modality against the events in the pattern.
If the *three conditions* are true, two types of result are produced, namely (1) the generation of probable meaningful combinations of data, and (2) the detected modality used by the user is correct. The generated meaningful data are stored in a database which will be used later by the fusion engine and at the same time serves as reference in case that such interaction is repeated in future. It is to be noted that the order of similarity matching is not important; it is possible that we verify the parameters ahead of the format or the events ahead of the parameters. However, if there is no match in any of these conditions, a feedback message is sent to the user for further action.

Step 5 – Fusion Engine: This module is responsible for the merging of data by gathering different input streams coming from various modalities and obtaining a meaningful combinational result. Fusion is a logical combination of two or more entities, in this case two or more modalities. Input signals are intercepted by the fusion agent and then combined taking into account some given semantic rules. These rules are presented by models, which are predefined patterns stored inside a knowledge base that describes one or many rules needed by the fusion engine for data merging.

2.3 Context Pattern

A *pattern* is defined as an idea that has been proven to be useful in one practical context and therefore will probably be useful in others [12]. Furthermore, patterns are often defined as something strictly described and commonly available. As stated earlier, patterns are predefined models that describe a modality; in our work, a pattern is composed of data format, events and parameters. Patterns are used to check a match of a user action involving a modality against the pre-defined data associated with modalities. Patterns are modeled, taking into account important characteristics, namely, the format of the pattern, its parameters and the modality events. Patterns are stored in a knowledge base. The meaning of each component in the pattern is as follows:

Parameters: These are the context parameters that affect a pattern; each modality is normally associated by some parameters. It is important to take these parameters into account in the design of patterns. Three types of parameters are essential:

- *User Context* - These are parameters that describe the status of the user as he works on a computing task. Sample user context parameters are the user preferences, profile, handicap, etc.
- *Environmental Context* – these are parameters that describe the user's working environment. Sample environmental parameters are noise level, darkness, etc.
- *System Context* - These are parameters that describe the status of the user's computing resources. Possible system context parameters are network, bandwidth, specifications, etc.

Format: The format of a pattern must contain the elements that suit a specific modality (entities, attributes, properties). These elements insure equivalence between the data of a modality and the pattern itself (e.g. starts time, end time of a modality, etc.).

Events: Each pattern is associated with specific predefined events. *Every modality* has its own set of events. For example, common events associated with the *touch screen modality* are as follows: Click on the screen, Touch screen lightly, Tap the screen once, Tap the screen twice, Keep a finger on an icon, Touch and drag an object on the screen. Indeed, the pattern that treats this modality (touch screen) must contain these events in the design.

2.4 Models of Rules

Another type of pattern is needed in our architecture, this pattern contains the Models of rules needed for data merging and it is stored inside a knowledge base. A multimodal fusion rule consists of constraints and result construction rules. Patterns are modeled, taking into account important characteristics, namely, the **Content** which specifies the semantic content of the model (e.g. object, type, location etc.). The **Category**, which indicates the category of the element (e.g. special gesture, command, etc.). The **constraints** which are the system, user and environment contexts that affect a rule. The **Time** which is a temporal interval that indicate when a rule can be applied. The **Probability** which define, how well an object fits to a category. The **Modality** which can be used by a rule.

3 Scenario

Assume that a regular user with no handicap is in his office. Using a laptop, he talks to his machine, saying “Draw a circle”. The modality recognition module of our architecture recognizes his speech and translates his words into a corresponding EMMA file. After such recognition, its data is interpreted by an interpretation module. Such interpretation is done by comparing the given data with the predefined patterns. The steps are as follows: First, a comparison is made between data format of the resulting EMMA file and the data format of the pattern by checking tags (e.g. comparing <*emma.start*> with pattern). If the pattern contains “*start*” tag and if all other tags in the EMMA file are presented, then the data format is checked and the condition is accepted, yielding a value for percentage of confidence. Next, a comparison of parameters will be made. Given that the user is a regular user and is in his office, such data is to be compared with the pattern. A predefined user context exists for a regular user, with no handicaps, and an environmental context being an office that is not noisy, having sufficient light, etc. and that the system context is set to normal. Then, the data collected by sensors in the office will be compared with the predefined parameters. The system descriptions will be compared against the system context in the pattern. Hence, if user context is equal to “no handicap” and data sensors detect that the user has no handicaps, then the first parameter is accepted. The same process will be done to check if environmental context is equal to “not noisy” and “there is light”, etc. Given that the sensors detect that there is no noise and there is light, then the second condition is accepted. If the system specification is equal to the given system context (e.g. computing network is available), then the third condition is accepted. Hence, the actual collected parameters and predefined pattern’s parameters are similar and therefore the percentage of confidence for this case is updated. The last comparison to be made is to find out if the event invoked by the user is similar to one of many events predefined in the pattern of a specific modality, in this case, speech. If a match is found, all the actual conditions are accepted and a final percentage of confidence is calculated. If this value is more than the specified level of confidence (e.g. 70%), we can conclude that a similarity between a modality and a pattern occurred and given data will be stored in the database. Now that data is stored in the database, the fusion agent will need models of rules to merge data, to do that, another comparison and similarity calculation is done to check matching between predefined models of rules stored inside a knowledge base and the command of a user, if there is similarity, the fusion agent will merge data according to the model of rules identified and a result of merged data is obtained that will lead to drawing of a circle. If not a feedback is generated.

4 Conclusion and future work

In this paper, we presented an architecture that is very useful in a multimodal system. The architecture proposes a solution that standardizes the interpretation of modalities. Such standardization is implemented by defining a set of patterns that are stored in a knowledge base. These patterns will be a reference that measures the similarity between data that was previously taken from different modalities and the pattern itself. This technique will offer the fusion engine the most meaningful combination of data that can be used in the fusion process. Future researches will includes more detailed information about the patterns and their design, by modeling a variety of them according to different modalities. Also a fusion engine that deal with these patterns will be designed and more scenarios will be proposed. The knowledge base will be more and more completed and at the end a prototype will be developed to test and validate the whole architecture.

Acknowledgement

This work has been made possible by the funding awarded by the Natural Sciences and Engineering Research Council of Canada (NSERC).

References

[1] P. C. Yuen, et al., Multimodal Interface for Human-Machine Communication vol. 48. Singapore: World Scientific Publishing Co., Pte. Ltd., 2002.

[2] B.-S. Shin, et al., "Wearable multimodal interface for helping visually handicapped persons," presented at the 16th international conference on artificial reality and telexistence, Hangzhou, China, 2006.

[3] R. Raisamo, et al., "Testing usability of multimodal applications with visually impaired children," IEE, Institute of Electrical and Electronics EngineersComputer Society, vol. 13, pp. 70-76, 2006.

[4] J. Lai, et al., "Examining modality usage in a conversational multimodal application for mobile e-mail access," International Journal of Speech Technology, vol. 10, pp. 17-30, 2007.

[5] M. Debevc, et al., "Accessible multimodal Web pages with sign language translations for deaf and hard of hearing users," presented at the DEXA 2009, 20th International Workshop on Database and Expert Systems Application, Linz, Austria, 2009

[6] Y. Li, et al., "An exploratory study of Web services on the Internet," presented at the IEEE International Conference on Web Services, Salt Lake City, UT, USA, 2007.

[7] R. P. Engel, Norbert. , "Modality Fusion.SmartKom: Foundations of Multimodal Dialogue Systems. Springer, Berlin," 2006.

[8] J. A. Norbert Reithinger, Tilman Becker, Anselm Blocher, Ralf Engel, Markus Löckelt, Jochen Müller, Norbert Pfleger, Peter Poller, Michael Streit, Valentin Tschernomas, "SmartKom - Adaptive and Flexible Multimodal Access to Multiple Applications, DFKI GmbH – German Research Center for Artificial Intelligence" 2003.

[9] M. R. Daniel Sonntag, "A Multimodal Result Ontology for Integrated Semantic Web Dialogue Applications Daniel Sonntag_, Massimo Romanelli DFKI GmbH. German Research Center for Arti_cial Intelligence Stuhlsatzenhausweg 3, d-66123 " 2006.

[10] B. Dumas, et al., "Description languages for multimodal interaction: a set of guidelines and its illustration with SMUIML," Journal on Multimodal User Interfaces, vol. 3, pp. 237-247, 2010.

[11] C. Desmet, et al., "<emma>: re-forming composition with XML," Literary & Linguistic Computing, vol. 20, pp. 25-46, 2005.

[12] M.Fowler, "Analysis Patterns – Reusable Object Models," 1997.

ABOUT THE AUTHOR

Ahmad Wehbi: PhD student, MMS Research Group, Université du Québec, École de technologie supérieure 1100, rue Notre-Dame Ouest, Montréal, Québec, H3C 1K3, LISV Laboratory, Université de Versailles-Saint-Quentin-en-Yvelines 10-12, avenue de l'Europe, 78140 Vélizy-Villacoublay, France, ahmad.wehbi.1@etsmetl.ca, Ahmad Wehbi@lisv.uvsq.fr

Amar Ramdane-Cherif: Phd Professor, LISV Laboratory, Université de Versailles-Saint-Quentin-en-Yvelines 10-12, avenue de l'Europe, 78140 Vélizy-Villacoublay, France, rca@prism.uvsq.fr

Chakib Tadj : Phd Professor, MMS Research Group, Université du Québec, École de technologie supérieure 1100, rue Notre-Dame Ouest, Montréal, Québec, H3C 1K3, Chakib.Tadj@etsmtl.ca