Multi-Dimensional Context-Aware Adaptation of Service Front-Ends

Project no. FP7 – ICT – 258030

Deliverable D.3.2.3

ASFE-DL: Semantics, Syntaxes and Stylistics (R3)

Due date of deliverable: 30/09/2013

Actual submission to EC date: 30/09/2013

Project co-funded by the European Commission within the Seventh Framework Programme (2007-2013)		
Dissemination level		
[PU]	**[Pubic]**	**Yes**

Document Information	
Lead Contractor	TID
Editor	CNR-ISTI
Revision	v.1.0 (20/09/2013)
Reviewer	UCL
Approved by	TID
Project Officer	Michel Lacroix

Contributors	
Partner	**Contributors**
ISTI	**Marco Manca, Fabio Paternò**

Changes			
Version	**Date**	**Author**	**Comments**
1	18/09/2013	ISTI	First document draft
2	23/09/2013	ISTI	Document for review
3	25/09/2013	UCL	First reviewed version
4	30/09/2013	ISTI	Final Version

Executive Summary

This report provides an update of the work in SERENOA for creating a model-based Description Language (DL) for User Interfaces (UI). In particular, it discusses an updated version of the Advanced Service Front End Description Language (ASFE-DL). Since in previous versions the Abstract level, and the concrete level for graphical interfaces were addresses, we have moved our attention to the vocal modality, which is acquiring an increasing importance also in mass market products. Thus, this document provides a first version of ASFE-DL that covers the Concrete Level for such vocal modality. The document ends with a set of conclusions and with a set of actions for further improving the set of languages composing ASFE-DL.

Table of Contents

1 Introduction

1.1 Objectives

In this document, we report on the updates to the second version of the Description Language (DL) for User Interfaces (UI) that it is used in Serenoa. The language is compliant with the Cameleon Reference Framework [Calvary2002]. Therefore, the UI modelling is addressed at various levels of abstraction. The previous version of the ASFE-DL only covered some updates of the Abstract User Interface (AUI) level and presented the Desktop Concrete level. In this document we define a first version of the ASFE-DL for the Vocal Concrete level.

1.2 Audience

Being a public deliverable, this document will be also available outside the project's consortium and is intended to be of interest to the following parties:

a) Members of the consortium, who will find here a detailed description of the fundamentals of the Description Language that the project is to use in the future.
b) Researchers and developers in the relevant fields: adaptation of SFEs, UI concepts, descriptive languages and medium-scale project software engineering.
c) EC officials that will use the information in this document as an account of the activities taken in the project tasks that inform this work.

1.3 Related documents

- **D3.1.3 Reference Models Specification (R3)** which provides an abstract model of the adaptation process, covering not only rules but also a model-based adaptation process.
- **D3.2.2 ASFE-DL: Semantics, Syntaxes and Stylistics (R2)** which describes the previous version of the ASFE-DL.
- **D3.3.2 AAL-DL: Semantics, Syntaxes and Stylistics (R2),** which specifies the language for expressing advanced adaptation logic (AAL). The actions described for adapting the User Interface may change properties of the UI at different levels of abstraction.
- **D4.5.3 Authoring Environment (R3)**, which provide the update on the development status of the authoring environment and analysis tools for the creating adaptive SFEs. This of course includes also creating ASFE-DL models.
- **D5.1.2 Serenoa Framework (R2**): which describes the Serenoa Framework, including the design of the common interfaces, the integration of the software components and the interface definitions and code documentation.
- **D6.2.2 Standardization Actions Report (Update 2)**, which describes the standardization effort on the W3C MBUI working group. The first version of the ASFE-DL language provided input for the standardization.

1.4 Organization of this document

After the introduction to the goals and the structure of the document, we recall the main concepts in abstract user interfaces in the ASFE-DL, and next we introduce the aspects that characterise the vocal modality.

We then detail the ASFE-DL concrete description for such modality, which shares the logical structure of the abstract level and provides refinements for the specific aspects characterising such modality. Lastly, we draw some conclusions and provide indications for future work.

2 ASFE-DL: Abstract UI level

2.1 Main Structure

The Abstract UI level describes the User Interface independently of the target platform and the interaction modality (graphical, vocal, etc.).

The element that represents an abstract user interface model is called *AbstractUIModel* and consists of a composition of *AbstractInteractionUnit*. Each AbstractInteractionUnit represents a part of an application user interface that should be presented to the user at once. An AbstractInteractionUnit is composed of *AbstractInteractors* and *AbstractRelationships.*

2.2 Connections

It is possible to model the navigation among the different abstract interaction units defining instances of the *Connection* class. A connection (which has a unique identifier, id attribute) specifies the target abstract interaction unit and the interactor (or interactors) that triggers such navigation. It is also possible for the connection to specify more than one target interaction unit. In this case, the specification will contain also the condition (see the related attribute in the specification of the Connection element) for the dynamic selection of the target among the specified set of abstract interaction units. In the previous deliverable (D.3.2.2) the back class has been added as a new type of connection: it connects the current AUI with the previously visited one.

2.3 Abstract Interactors

An Abstract Interactor represents a generic user interface object and is composed of a set of different elements, according to the interaction semantics:

- *Selection*: This abstract class represents the possibility to select one or multiple values (the refinement is called *SingleChoice* and *MultipleChoice* respectively) from a predefined set of choice (*choiceElement*);
- *Edit*: This class represents interactors that allow the input of manually edited values;
- *OnlyOutput*: This class represents interactors that show information to the user;
- *Control*. This abstract class represents interactors that allow the user to perform actions, submit data or navigate the user interface. The refinements *Activator* and *Navigator* distinguish the objects mainly devoted to performing actions from the ones to perform UI navigation.

Each *AbstractInteractor* can be optionally connected to a *DataItem*, specifying that it represents the referenced data item in the user interface.

2.4 Abstract Relationships

The *AbstractRelationship* abstract class represents the base type for all the possible logical connections among the interactors. An abstract relationship consists of a composition of other abstract relationships and abstract interactors. It is possible to specify the following relationship types:

- *Grouping*: this class represents a generic group of interactors, which share a logical connection
- *Ordering*: this class represents a set of interactors that have an ordering relation among the elements. There also is *the ordering_value* attribute, which represents the value for an ordered presentation of the content
- *Hierarchy*: this class represents a set of interactors where different levels of importance can be identified. There is also the *hierarchy_value* attribute, which is used for providing a hierarchical presentation of the content.
- *Repetition*: this class represents a template for a list of interactors that have to be repeated in order to represent a dynamic list of items, coming from a data source.
- *Dependency*: A dependency is used when we want to model a dependency relation between 1

interactor/interactor group and other N interactors/interactor groups.

2.5 External Model

An AbstractInteractionUnit can be connected to an *ExternalModel* in order to define references to models that are needed by the AUI, but that are not defined into the AUI itself; the external model can be refined in Data Model and Context Model whose values influence both the rendering and the behaviour of a UI, but that can be managed separately with respect to the UI model.

2.6 External Functions

*The ExternalFunction*s element defines a list of functionalities, external to the modelled application, which are exploited inside the model. Since we are defining a language for Service Front Ends, the external function class is refined into *SoapReference* and *RestReference:* each one contains a set of attributes that allows the identification of the operations according to the represented technology.

3 The Vocal Modality

The vocal modality is acquiring an increasing diffusion since its underlying technology has fast been improved in recent years. It is now used in various contexts: mobile (e.g. the iPhone SIRI environment), desktop, and web applications. In the latter case, the vocal modality allows users to interact and browse web applications through voice and keypads. Thus, the web becomes more than just the web pages we can see, but also becomes the web pages we can hear and speak to [Kazuyuki2013]. Regarding this modality, W3C has developed a suite called Speech Interface Framework in which one of the main contribution is VoiceXML, an XML language for writing Web pages you interact with by listening to spoken prompts and jingles, and control by means of spoken input [Ragget2001].

Vocal interaction is very different from graphical interaction, in vocal interfaces users can hear only one thing at a given time. The application and user take it in turns to speak: the application prompts the user, and the user in turn responds. Thus, it is a modality that requires interactions more sequential than the visual one.

Vocal interaction is useful in several contexts: when the visual channel is busy (e.g. while driving), for disabled people or while the user is moving. The vocal features make it suitable to support quick access to information and to interact in a way more similar to that used for communication among humans.

Because the vocal interaction is different from the graphical one, some specific interaction techniques have been developed for this modality:

- **Barge-in** technique allows users to interrupt the speech synthesizer by using speech or DTMF (Dual-Tone Multi-Frequency) input; this speeds up conversations but is not always desired. If the application author requires that the user must hear all of a warning, legal notice, or advertisement, barge-in should be disabled;
- **Tapered prompting** technique: tapered prompts are those that may change with each attempt. Information-requesting prompts may become terser under the assumption that the user is becoming more familiar with the task [McGlashan2004]. This technique allows you to give progressively more detailed prompts when the user is having difficulty in answering, or, prompts can change just to make the interaction more interesting.

One of the main challenges of vocal interaction is error recognition; the errors can be divided into three categories:

1. Rejection errors: when the recognizer does not match the user input with any expected utterance;
2. Substitution errors, when the platform recognize a wrong but legal input;
3. Insertion errors: if the recognizer accepts a noise as a valid input.

If one of these types of errors occurs, it should be possible to give a message to the user and to repeat the next (and more detailed) input. To reduce the occurrence of substitution and insertion errors we allow designers to verify the utterance when necessary (see the ask confirmation attribute described in paragraph 4.4.2). To manage rejection errors we defined the no match event (see paragraph 4.2).

4 ASFE-DL: Vocal Concrete Level

The definition of a Concrete Language consists mainly on the refinement of the abstract classes with the elements that can be used in the specific platform in order to build the UI.

In the previous deliverable version we introduced the graphical Desktop Concrete Version of the ASFE-DL language; in this deliverable we present the refinements of the abstract interactors and relationship that are possible to use in the Vocal platform.

4.1 Vocal Type Element

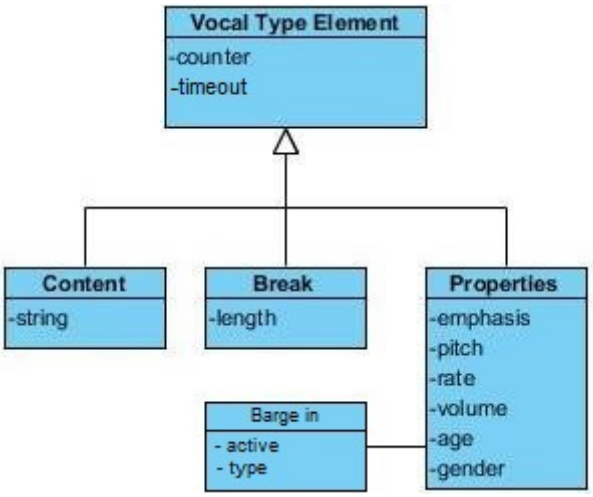

Figure 1: Vocal Element

Figure 1 shows the UML diagram for the vocal type element; this is the most important element, it is used as a prompt and feedback inside concrete vocal interactors and as speech element inside only output elements. The content element defines the text that the vocal platform must synthesize or the path where the platform can find the text resources. The break elements permit to insert a break after synthesizing the text. Through the vocal properties element it is possible to define properties of the synthesized voice, such as emphasis, pitch, rate, volume, age and gender. The Barge-in element, defined as a child of properties element, implements the barge in technique described in the previous chapter; it has an boolean active attribute and a type attribute that describes how the synthesizer react to a barge in interrupt call. The possible values of the type attribute can be:

- **Speech**: the synthesizer immediately stops when detects an user input;
- **Hotword**: the synthesizer stops only when recognize a specified word.

Through counter attribute we provide designers with the possibility of implementing the tapered prompting technique: in this way the messages for the user become more explicit when the number of errors increase. This is obtained through multiple prompts with sequential counter values: if after the first prompt the user does not insert an input the vocal platform will synthesize the prompt with the following counter value. In paragraph 4.4.2 we show an example of how the barge-in and the tapered prompt techniques are defined in ASFE-DL. The timeout attribute defines the time interval before a no input event is generated.

4.2 Vocal events

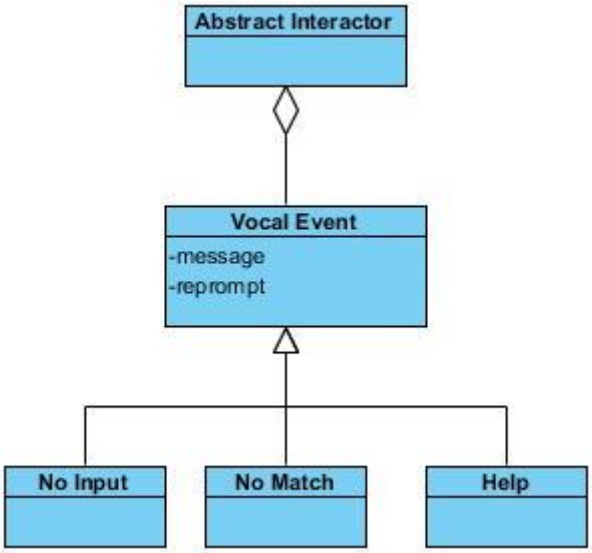

Figure 2: Vocal Event

While in graphical interfaces the events are related mainly to mouse and keyboard activities, in vocal interfaces we have to consider different types of events (see Figure 2): 'no input' (the user has to enter a vocal input but nothing is provided within a defined amount of time), 'no match', the input provided does not match any possible acceptable input, and help, when the user asks for support (in any platform specific way) in order to continue the session. All of them have two attributes: message, indicating what message should be synthesized when the event occurs, and re-prompt, to indicate whether or not to synthesize the last communication again.

4.3 Grammars

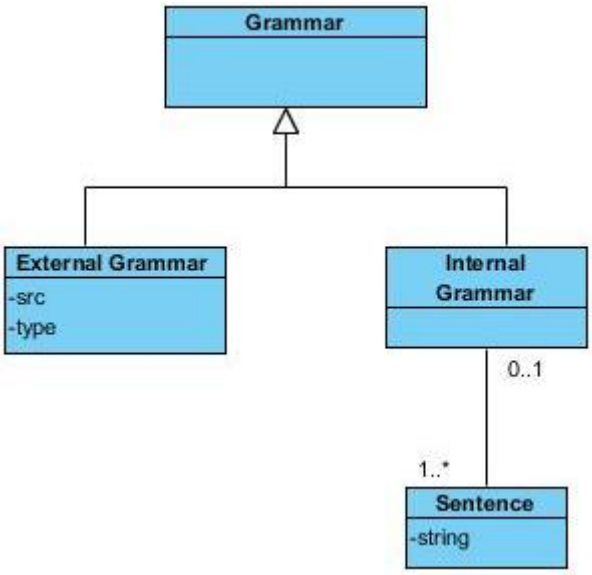

Figure 3: Vocal Grammar

For an automatic speech recognition (ASR), it can be problematic to recognize a speech input, for this reason we provide the possibility of specifying a grammar element (Figure 3) in which it is possible to indicate the possible inputs.

It is possible to define an external grammar element indicating the URL of the document containing the grammar; otherwise it is possible to define an internal grammar listing all the possible input sentences.

4.4 Concrete Vocal Interactors

4.4.1 Control

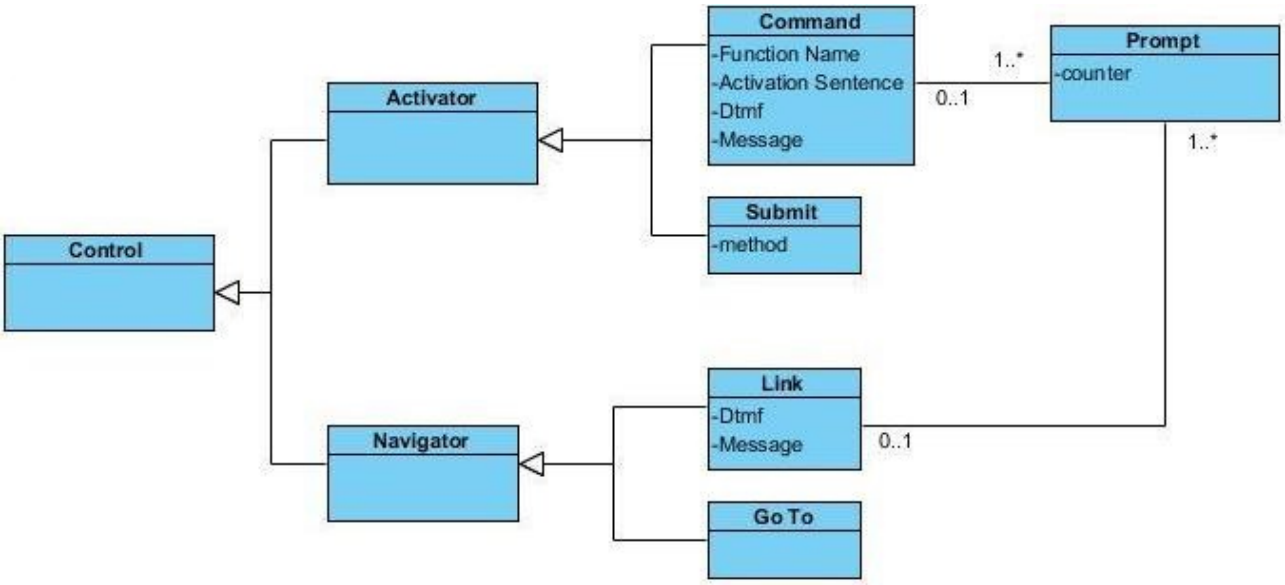

Figure 4: Control Interactors refinement

The concrete refinements for the *Activator* are *Command* and *Submit*; while the refinements for the *Navigator* are *Link* and *GoTo (*Figure 4*)*.

The Activator Command refinement is useful to execute a script, it is possible to render a vocal prompt to the user in order to synthesize a sentence explaining the command and how to activate it.

The possible attributes are:

- Function name: name of function previously defined in the document;
- Activation sentence: Sentence to say to activate the activator;
- Dtmf: dtmf to digit to activate the activator;
- Message: the message that will be synthesized when the command is activated.

The Submit element sends a set of data previously collected through the vocal platform to a server; the server URL is specified inside the connection target element defined before. These elements have only an attribute that indicates the method (GET or POST) used to send the data.

Regarding the navigator refinements, the GoTo changes automatically the vocal interaction unit (the target vocal interaction unit is linked to the navigator through the connection target element with the same id of the navigator). The link refinement is used for user-triggered change of presentation, inside this kind of element it is possible to add a vocal type element named prompt that is a sentence to explain to the user how to navigate to the next interaction unit. The link attributes (message and dtmf) have the same purpose explained before.

4.4.2 Edit

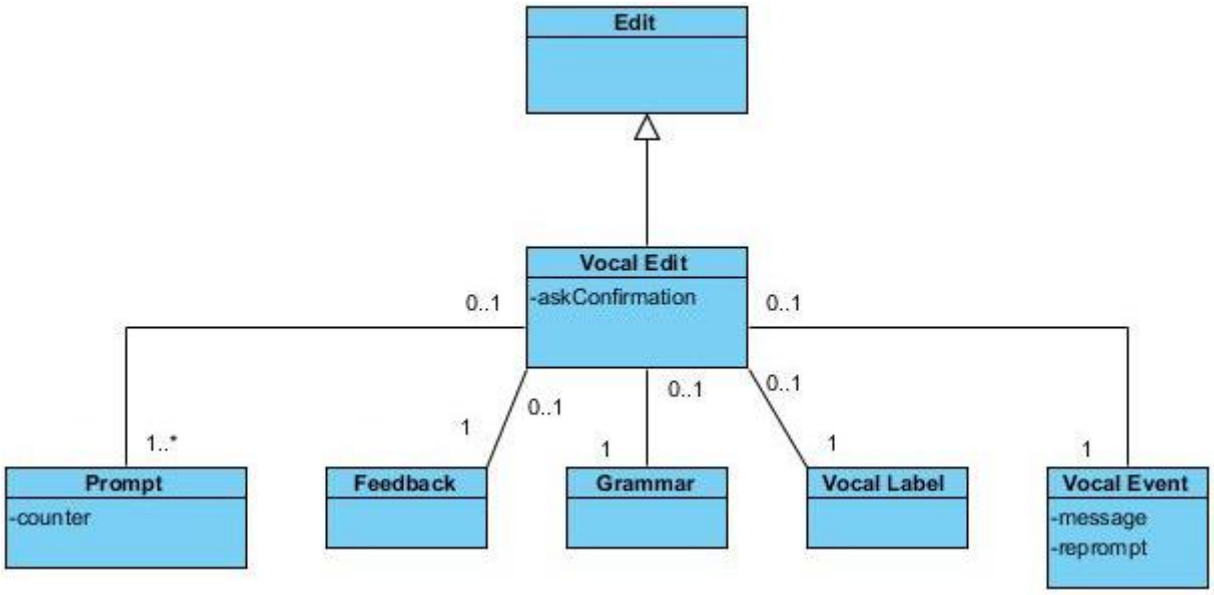

Figure 5: Edit refinement

The edit element gathers a user input; from the graphical point of view the edit element is an editable textual field, in the vocal context we refined this abstract element with the vocal edit element (Figure 5) which permits setting multiple vocal prompt (that is a question to ask to the user explaining the input needed), a vocal feedback (vocal output before re-prompt the last input acquired) and specifying a grammar (internal or external) for the recognition of the user input. Moreover it is possible to specify a vocal label that is a sentence introducing the current value (if present). The edit element has the askConfirmation boolean attribute to specify if the platform has to ask a confirmation of the inserted input.

In Figure 6 we show the definition of an edit element in ASFE-DL vocal concrete language, which also shows an example of tapered prompt and barge in.

```
<edit>
    <vocal_edit askConfirmation="true">
        <prompt counter="1" timeout="5s">
            <content string="Please, insert your age"/>
            <break length="1s"/>
            <properties volume="50" gender="male">
                <bargein active="true" type="speech"/>
            </properties>
        </prompt>
        <prompt counter="2" timeout="10s">
            <content string="Please, insert your age, the input expected is a number"/>
            <break length="1s"/>
            <properties volume="50" gender="male">
                <bargein active="false"/>
            </properties>
        </prompt>
        <feedback>
            <content string="Your age is"/>
            <break length="1s"/>
        </feedback>
        <vocal_events>
            <noinput message="You did not insert an input, try again" reprompt="true"/>
            <nomatch message="You did not insert a valid input, try again" reprompt="true"/>
        </vocal_events>
    </vocal_edit>
</edit>
```

Figure 6: Vocal edit: tapered prompt and barge in implementation

4.4.3 Only Output

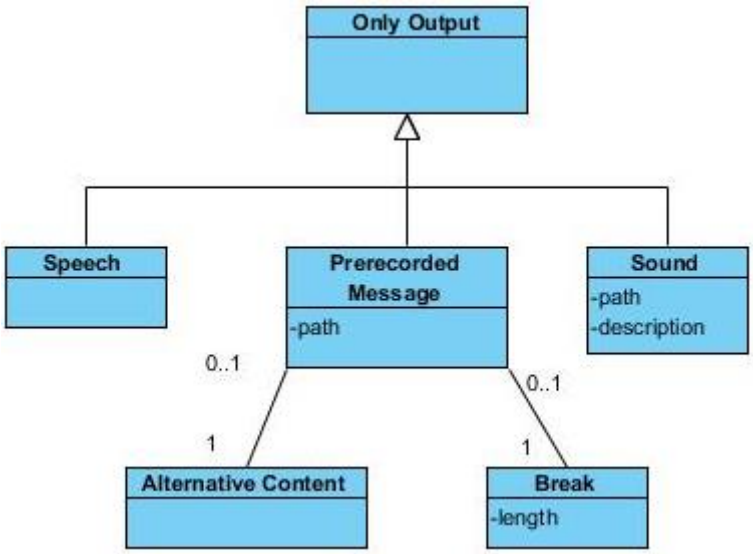

Figure 7: Only output element

The list of the concrete refinements of the Only Output elements (depicted in Figure 7) is:

1. Speech: this is a vocal type element described before;
2. Pre-recorded Message: it defines the path of pre-defined audio resources that must be played. We support the case of missing resources by defining an alternative content that can be synthesized when this case occurs. It also possible to add a break after playing the message
3. Sound: this element permits defining the path of a non-vocal audio resource that must be played by the platform. It is also possible to insert a textual description of the sound that could be used as additional information regarding the sound content.

4.4.4 Selection

Selection interactors allow users to select a value between a set of choice elements. The abstract level distinguishes this kind of interactors between single and multiple choices.

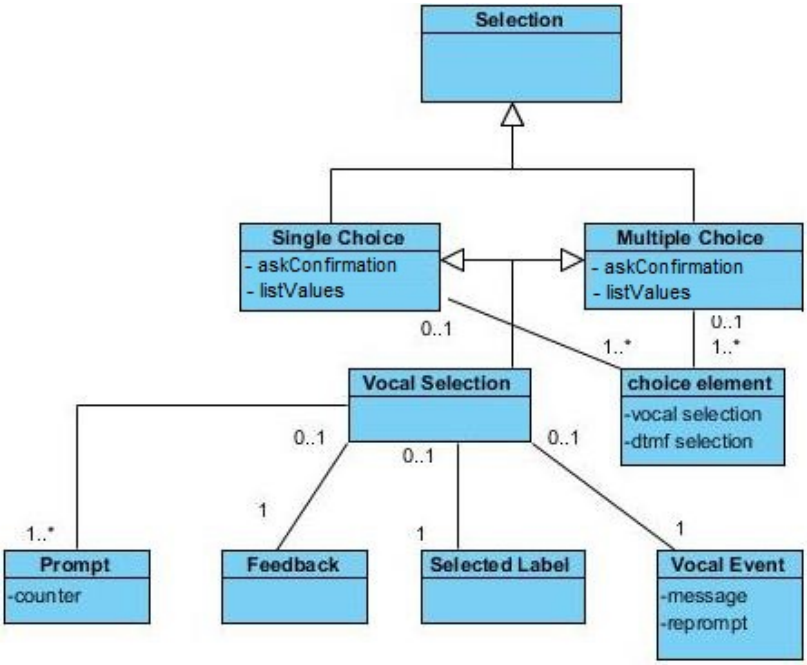

Figure 8: Selection refinement

To support vocal interaction we defined the vocal selection (Figure 8) type that contains vocal prompt(s) to ask a question(s) to the user about the input required, a feedback that is the output for the users before re-prompt the input acquired. It is possible to define vocal input (word or sentences) or DTMF input. The selected label element represents a sentence that introduces the current selected value(s). Vocal selection element has two attributed: askConfirmation (boolean attribute) to specify if the platform has to ask a confirmation of the inserted input and list values (boolean) that indicates if the platform has to list all the possible vocal choices. Depending on the type of selection one or multiple elements can be selected, semantic differences between these two interactors are introduced at abstract level.

4.4.5 Concrete Vocal Relations

Figure 9 shows the UML class diagram of the concrete refinements of the AbstractRelation classes in the vocal concrete platform.

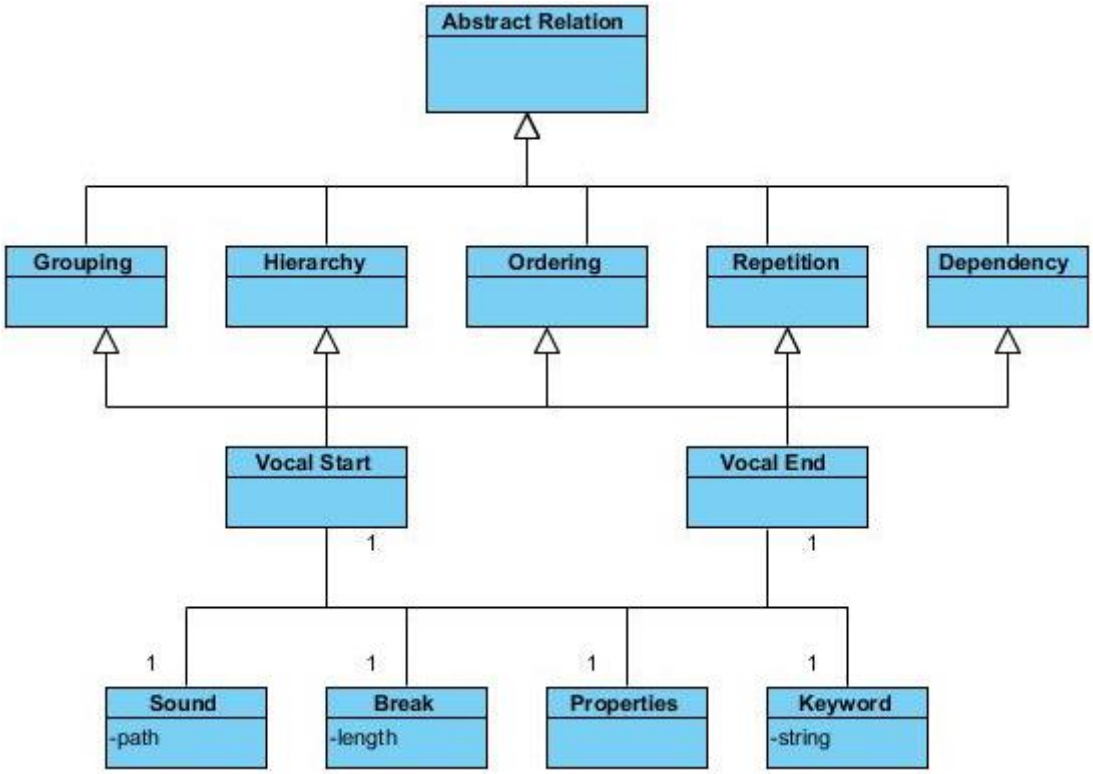

Figure 9: Relation refinement

4.4.6 Grouping

The grouping element permits the composition of elements, from a graphical point of view we can identify this element as a container of graphical elements; for the vocal interface we decided to allow users to specify the start and the end of a grouping through inserting a sound or inserting a pause. Other solutions can be change the vocal properties (voice gender, volume, etc.) or insert a keyword that explicitly define the start and the end of the grouping.

4.4.7 Hierarchy

The hierarchy represents a set of interactors where can be identified different levels of importance. It is possible to identify the beginning and the end of this element in the same way of grouping elements.

4.4.8 Ordering

This element represents a set of interactors that have an ordering relation among elements. It is possible to identify the beginning and the end of this element in the same way of grouping elements.

4.4.9 Repetition

It represents a template for presenting the generic item of a list, its content will be repeated at run time. It is possible to identify the beginning and the end of this element in the same way of grouping elements.

4.4.10 Dependency

It is used when we want to model a dependency relation between one interactor/interactor group and other interactors. It is possible to identify the beginning and the end of this element in the same way of grouping elements.

5 Conclusions

5.1 Summary

In this deliverable we have presented the refinement of the ASFE-DL Abstract level for describing vocal concrete interfaces. This shows that the ASFE-DL approach proposed is not limited to the graphical modality. In addition, it is also possible to extend such approach to obtaining versions that are able to support combinations of such modalities at the same time.

5.2 Future Work

Future work will be dedicated to extending the ASFE-DL to other modalities. For example, the gestural modality can be an important aspect to consider. We also plan to investigate how to describe interfaces combining various modalities.

This work can also be a useful input for the W3C working group on model-based user interfaces. So far such group has only considered the task and the abstract level. Thus, input on how to formalise the description of user interfaces exploiting the vocal modality can help to extend the scope of such standardization activities.

6 References

[Calvary2002] The CAMELEON Reference Framework, G. Calvary, J. Coutaz, D. Thevenin, L. Bouillon, M. Florins, Q. Limbourg, N. Souchon, J. Vanderdonckt, L.Marucci, F.Paternò, and C.Santoro, CAMELEON Project, September 2002.

[Kazuyuki2013] Voice Browser Activity (W3C), http://www.w3.org/Voice/, Kazuyuki Ashimura

[McGlashan2004] Voice Extensible Markup Language (VoiceXML) Version 2.0 http://www.w3.org/TR/voicexml20/ S. McGlashanm D.C. Burnett, J. Carter, P. Danielsen, J. Ferrans, A. Hunt, B. Lucas, B. Porter, K. Rehor, S. Tryphonas

[Paternò2010] Deriving vocal interfaces from logical descriptions in multi-device authoring environments Paternò F. Sisti C. - In: ICWE 2010 - 10th International Conference on Web Engineering (Vienna, Austria, 5-9 July 2010). Proceedings, pp. 204 - 217. (Lecture Notes in Computer Science, vol. 6189). Springer Berlin, 2010

[Ragget2001] Getting started with VoiceXML 2.0 http://www.w3.org/Voice/Guide/

Acknowledgements

- TELEFÓNICA INVESTIGACIÓN Y DESARROLLO, http://www.tid.es
- UNIVERSITE CATHOLIQUE DE LOUVAIN, http://www.uclouvain.be
- ISTI, http://giove.isti.cnr.it
- SAP AG, http://www.sap.com
- GEIE ERCIM, http://www.ercim.eu
- W4, http://w4global.com
- FUNDACION CTIC http://www.fundacionctic.orglossary
- http://www.serenoa-fp7.eu/glossary-of-terms