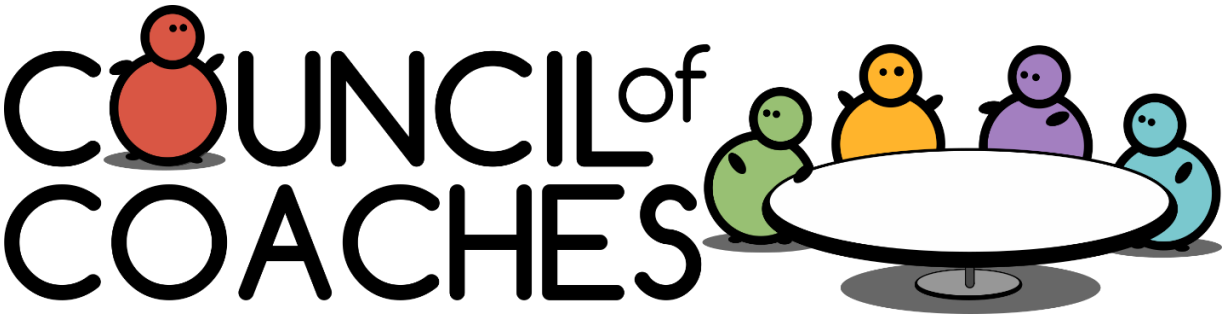

D6.1: Requirements and Concepts for Interaction Mobile and Web

Dissemination level: Public

Document type: Report

Version: 1.0.1

Date: February 28, 2018 (original)

March 5, 2019 (this version)

This project has received funding from the European Union's Horizon 2020 research and innovation programme under Grant Agreement #769553. This result only reflects the author's view and the EU is not responsible for any use that may be made of the information it contains.

Document Details

Project Number	769553
Project title	Council of Coaches
Title of deliverable	Requirements and Concepts for Interaction Mobile and Web
Due date of deliverable	February 28, 2018
Work package	WP6
Author(s)	Catherine Pelachaud (UPMC), Reshmashree Bangalore Kantharaju (UPMC), Randy Klaassen (CMC), Merijn Bruijnes (CMC)
Reviewer(s)	Randy Klaassen (CMC), Gerwin Huizing (CMC)
Approved by	Coordinator
Dissemination level	PU: Public
Document type	Report
Total number of pages	31

Partners

- University of Twente – Centre for Monitoring and Coaching (CMC)
- Roessingh Research and Development (RRD)
- Danish Board of Technology Foundation (DBT)
- Sorbonne University (SU)
- University of Dundee (UDun)
- Universitat Politècnica de València, Grupa SABIEN (UPV)
- Innovation Sprint (iSPRINT)

Abstract

The goal of this Work Package (WP6) is to design, implement, and evaluate the Human Computer Interaction aspects of the Council of Coaches. In this deliverable, we provide a detailed description of existing platforms and tools that we intend to use in the Council of Coaches project. We also provide a short overview of existing agent-based user interfaces that the Council of Coaches might build on top of.

Corrections

v1.0.1	Correctly applied EU logo on header page.
	Changed UPMC to Sorbonne University (SU).

Table of Contents

1 Introduction 7
2 Objectives 8
3 SAIBA Framework 9
3.1 Function Mark-up Language (FML) 10
3.2 Behaviour Mark-up Language (BML) 10
3.3 Behaviour Lexicon 11
4 Reference Architecture Council of Coaches 13
5 Agent Platforms 14
5.1 GRETA 14
5.1.1 FML-APML, Affective Presentation Mark-up Language, Function Mark-up Language 14
5.1.2 Communicative intentions in FML-APML 15
5.1.3 Attributes of FML-APML tags 16
5.1.4 Emotion tag 16
5.1.5 Speech and synchronisation 17
5.2 ASAP - Articulated Social Agents Platform 18
5.2.1 ASAP Multi-Agent Extension 18
5.2.2 UMA: Unity Multipurpose Avatar 19
6 Tools 21
6.1 Editors 21
6.2 Integration of Greta platform within Unity 3D 23
6.3 Creation of new characters for the Greta platform 23
7 Home and Mobile UI 24
8 Conclusion 28
9 Bibliography 29

List of figures

Figure 1: The SAIBA framework for multimodal behaviour generation, showing how the overall process consists of three sub-processes at different levels of abstraction. 9
Figure 2: Example of a BML request indicating a speech element starting when the specified gaze behaviour is ready to be displayed. 11
Figure 3: Example of a gesticon pair: nonverbal behaviour specification for the backchannel agreement. 11
Figure 4: Technical component architecture from the EU H2020 project ARIA VALUSPA. 13
Figure 5: Example of an FML-APML request indicating the speech block specified using the SSML standard and communicative intentions. 15
Figure 6: First demo of the Multi-agent extension of the ASAP BML realizer tool. 19
Figure 7: Agent created with UMA which can be controlled by ASAP. 20
Figure 8: Facial expression created using MPEG-4. 21
Figure 9: Facial Expression created with Action Units. 22
Figure 10: Editors to create gesture and hand shape. 22
Figure 11: The Smarcos coaching system, including "BML-enabled" devices where BML driven agents are present. 24
Figure 12: A coaching message presented by the ASAP PictureEngine agent on an Android device. 25
Figure 13: A coaching message presented by the ASAP 3D agent on a PC. 25
Figure 14: A coaching message presented by a virtual human and in text on 2 different devices (a mobile Android device and Windows PC). 26
Figure 15: A coaching message from the PERGAMON system (in Dutch). 26

Symbols, abbreviations and acronyms

ASAP	A Social Agent Platform
AU	Action Unit
BML	Behaviour Mark-up Language
CMC	Centre for Monitoring and Coaching
COUCH	Council of Coaches
D	Deliverable
DBT	Danish Board of Technology Foundation
EC	European Commission
ECA	Embodied Conversational Agent
ECC	Embodied Conversational Coach
FACS	Facial Action Coding System
FAP	Facial Animation Parameter
FML	Function Mark-up Language
FML-APML	FML - Affective Presentation Mark-up Language
ISPRINT	Innovation Sprint
M	Month
MS	Milestone
RRD	Roessingh Research and Development
SAIBA	Situation, Agent, Intention, Behaviour, Animation
SU	Sorbonne University
TCP-IP	Transmission Control Protocol – Internet Protocol
UDun	University of Dundee
UMA	Unity Multipurpose Avatar
UPV	Universitat Politècnica de València
UT	University of Twente
WP	Work Package
XML	Extensible Mark-up Language

Glossary

Behaviour Mark-up Language, BML	An XML-like mark-up language especially suited for representing communicative behaviour.
Coach	An entity that interacts with a person and in doing so applies coaching strategies. It can be a virtual human avatar or software-based.
Embodied Conversational Agent, ECA	A virtual or robotic human-like character that demonstrates many of the same properties as humans in face-to-face conversation, including the ability to produce and respond to verbal and nonverbal communication.
Embodied Conversational Coach, ECC	An ECA that takes the role of a Coach.
Function Mark-up Language, FML	An XML-like mark-up language, especially suited for representing communicative intentions.
SAIBA: Situation, Agent, Intention, Behaviour, Animation	A common framework for multimodal generation for ECA.
Multimodal Behaviour	Communicative and emotional signals conveyed through different channels such as facial expression, gaze, body movement, gesture, prosody.
Communicative Intention	The message/idea the speaker aims to convey through verbal and/or nonverbal means.

1 Introduction

The Council of Coaches project aims to develop a tool to provide virtual coaching for ageing people to improve their physical, cognitive, mental and social health. The project combines state-of-the-art 3D virtual characters with language and reasoning technology and applies this to the area of lifestyle and behaviour change coaching. The council consists of a number of Embodied Conversational Coaches (ECCs), each specialised in their own specific domain. They interact with each other and with the user to inform and motivate them, and discuss issues related to their health and well-being. Six different expert coaches will be developed with knowledge in specific domains. They will be designed to have different physical appearances, roles and use different persuasive strategies i.e., emotional or rational. This work will make use of and build upon the GRETA/VIB platform developed at UPMC (Pecune, Cafaro, Chollet, Philippe, & Pelachaud, 2014) for multimodal behaviour generation and for visualising Embodied Conversational Agents (ECA). The coaches will respond in a highly personalised manner based on the holistic behaviour modelling and analysis component which will detect short-term behaviour events and long-term behaviour trends using sensors (on-body and off-body sensing). A multi-party Dialogue and Argumentation framework will be developed which will provide the dialogue actions to be communicated by the virtual coaches.

The main objective of this work package is to design, implement and evaluate the user interaction in different use case scenarios. The coaches will be designed to have different physical appearances, roles and use different persuasive strategies. The coaches will also be modelled to handle turn-taking in multi-party conversations and maintain user engagement.

The aim of this deliverable is to present the initial specifications, the platforms to be used and their requirements and an overview of the existing tools that are relevant to the Council of Coaches project.

2 Objectives

The objective of this deliverable is to provide an overview of existing works on which future developments within the Council of Coach project will rely on. These existing works concern overall human-agent architecture, virtual agent platform, tools to design virtual agent's characteristics and behaviours. This deliverable defines the initial Human-Computer Interaction component design and the agent platforms and their requirements. In this report, we will focus on providing a detailed explanation of the research objectives of WP6 and detailed explanations of the technologies and tools intended to be used.

This aims of this deliverable are:

- To present the existing agent platforms that will be used for the Council of Coaches project;
- To present the existing User Interfaces relevant to the Council of Coaches project;
- To present the tools used for creating behaviours;
- To explain the representational languages used for controlling the agent on functional and behavioural level;
- To propose the overall human-agent reference architecture.

In Section 3, we present the SAIBA framework which is the standard architecture common to both the agent platforms i.e., GRETA and ASAP. An overall architecture of the ARIA project that can be extended to a multi-agent version for the Council of Coaches project is presented in Section 0. In Section 5, we provide a detailed explanation about the languages and extensions specific to the agent platforms mentioned earlier and present the tools available in Section 0. Finally, we present the existing User Interfaces relevant to the project in Section 0.

3 SAIBA Framework

Virtual agents are able to communicate verbally and nonverbally with human users and/or other virtual agents. Given a set of intentions and emotions to be communicated, the platform instantiates them into sequences of synchronized nonverbal behaviours. It can be used to compute the multimodal behaviours when the virtual agent acts as a speaker or as a listener.

Let us first present the overall architecture of embodied conversational agents.

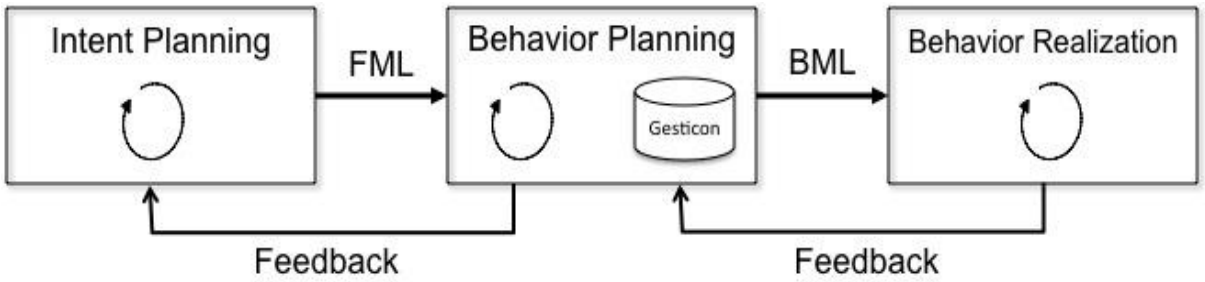

Figure 1: The SAIBA framework for multimodal behaviour generation, showing how the overall process consists of three sub-processes at different levels of abstraction.

The global architecture of the system, which is a SAIBA compliant architecture, is depicted in Figure 1. SAIBA is a common framework for the autonomous generation of multimodal communicative behaviour in embodied conversational agents (ECA) (Kopp, et al., 2006). SAIBA stands for Situation, Agent, Intention, Behaviour, Animation. This framework divides the overall behaviour generation process into three sub-processes, each bringing the level of communicative intent closer to actual realisation through the agents' embodiment (Vilhjálmsson H. H., 2009). SAIBA has adopted the strategy of using separate interfaces to specify an agent's communicative function and its communicative behaviour at two levels of abstraction, where the functional level determines the intent of the agent, that is, what it wants to communicate, and the behavioural level determines how the agent will communicate by instantiating the intent as a particular multimodal realisation. This separation can be seen as two independent components where one component represents the mind of an agent and the other component represents the body. During a user-agent dyadic interaction, for example, the agent's mind decides what function to accomplish (e.g. greeting), while the body receives what the mind decides to communicate and renders it at the surface level, according to available communication channels and capabilities of the agent.

This design strategy has several advantages. First, the agent's mind can produce decisions and intents independently of the body, so for example, the same mind can be used for a different agent's embodiment (e.g. virtual vs. robotic) or shared across systems. Second, the same communicative function can be delivered with different surface forms (i.e. verbal or nonverbal behaviour in case of ECAs) depending on the mental state of the agent or intended attitude that the agent aims to show off towards the user. Thus, an agent that wants to express a friendly attitude towards the user might accomplish the same function (e.g. greeting) by using different nonverbal behaviours compared to another agent that aims to express hostility.

This strategy also allows for sharing and reusing existing working components to speed up the process of getting full conversational systems up and running. For example, the same SAIBA platform can use the ASAP realizer or the Greta-Behaviour Realizer without changing its other modules. Similarly, the intent planner can be linked to a dialogue manager like the Dialogue and argumentation framework

developed by the University of Dundee (WP5) or to other dialogue managers (e.g. Flipper developed by University of Twente or DISCO developed by Sidner (Sidner & Rich, 2012)).

The interfaces connecting the components of the SAIBA platform are both at the high level, between intent planning and behaviour planning, and at the lower level in another interface, between behaviour planning and behaviour realisation. They are called Function Mark-up Language (FML) (Cafaro, et al., 2014) (Heylen et al., 2008) and Behaviour Mark-up Language (BML) (Kopp, et al., 2006) (Vilhjálmsson, et al., 2007) respectively. These languages are designed to be:

- Independent of a particular application or domain.
- Independent of the employed graphics and sound player model.
- To represent a clear-cut separation between information types (function-related versus process-related specification of behaviour) (Kopp, et al., 2006).

The framework also presents a Behaviour Lexicon (Gesticon), that can be used by the Behaviour Planner. This Gesticon is a dictionary which can contain predefined BML behaviour definitions. Currently BML has been standardised to a first official version adopted by international researchers, however a unified language specification for FML is still a work in progress (c.f. (Cafaro, Vilhjálmsson, Bickmore, Heylen, & Pelachaud, 2014)).

3.1 Function Mark-up Language (FML)

FML should describe communicative and expressive functions without any reference to physical behaviour, representing in essence what the agent's mind decides. It is meant to provide a semantic description that accounts for the aspects that are relevant and influential in the planning of verbal and nonverbal behaviour. An FML description must thus fulfil two tasks. Firstly, it must define the basic semantic units associated with a communicative event. Secondly, it should allow the annotation of these units with properties that further describe communicative function such as expressive, affective, discursive, epistemic, or pragmatic functions (Cafaro, Vilhjálmsson, Bickmore, Heylen, & Pelachaud, 2014) (Kopp, et al., 2006).

3.2 Behaviour Mark-up Language (BML)

BML describes the behaviours to express given a function, therefore multimodal behaviour should be described so that it can be used to control an agent (Vilhjálmsson, et al., 2007). The last stage handles the realisation of the behaviour by interpreting the incoming BML and making sure the virtual character behaves accordingly. The behaviour realisation depends on the particular realisation model and can be very diverse. Animations, for example, can be procedural or fixed and chosen from a repository. Sounds can be generated by a text-to-speech engine or played from a file. Therefore, what is specified by BML is independent of any specific realisation method (Kopp, et al., 2006) (Vilhjálmsson, et al., 2007).

The BML language allows us to specify the signals that can be expressed through the agent's communication modalities. Each BML top-level tag corresponds to a behaviour the agent is to produce on a given modality: head, torso, face, gaze, body, legs, gesture, speech, lips. Each BML tag also contains temporal information that corresponds to the timing of appearance and ending of the signals. A Behaviour Realizer generates the animation that is described in the BML message. An example of a BML excerpt is shown in Figure 2. This example shows how a gaze behaviour is synchronised with the start of the speech whose text is given within the label <text>.

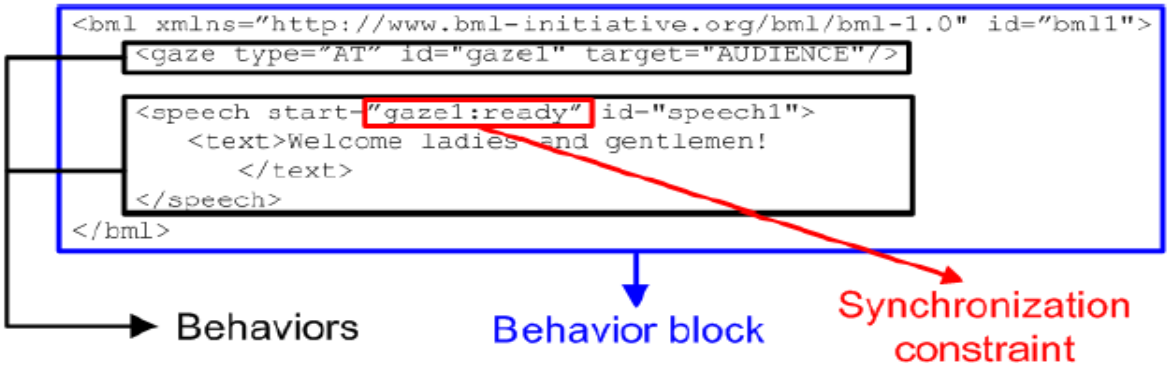

Figure 2: Example of a BML request indicating a speech element starting when the specified gaze behaviour is ready to be displayed.

The emotional state tags give the possibility to specify an intensity (as a numeric parameter from 0 to 1) and a regulation type, that controls for felt, faked (emotion aimed at simulating) and inhibited emotions (felt but aimed at being inhibited by the agent). The world tag makes it possible to reference to entities in the world and their properties (physical or abstract) (Mancini & Pelachaud, 2008).

3.3 Behaviour Lexicon

A Behaviour Lexicon contains pairs of mappings from communicative intentions to multimodal signals. The internal behaviour realizer instantiates the corresponding multimodal behaviours, it handles the synchronisation with speech and procedurally generates animations for the ECA. Figure 3 shows an example of a pair of communicative intentions and its corresponding behaviour realisations. The provided example illustrates how a backchannel to mean agreement can be expressed through head nod with different amplitudes and with different facial expressions (raised eyebrow or various types of smile). Each alternative signal is linked to a probability of selection.

```
<behaviorset name="backchannel-agreement">
    <signals>
        <signal id="1" name="Nod_Middle" modality="head">
            <alternative name="Nod_Big" probability="0.3"/>
            <alternative name="Nod_Small" probability="0.2"/>
        </signal>
        <signal id="2" name="faceexp=Liking" modality="face">
            <alternative name="faceexp=raise_brows" probability="0.2"/>
            <alternative name="faceexp=Smile_Small_Closed" probability="0.2"/>
            <alternative name="faceexp=Smile_Small_Open" probability="0.2"/>
        </signal>
        <signal id="3" name="gaze=look_at" modality="gaze"/>
    </signals>
    <constraints>
        <core>
            <item id="1"/>
        </core>
        <rules>
            <implication>
                <ifpresent id="2"/>
                <thenpresent id="3"/>
            </implication>
        </rules>
    </constraints>
</behaviorset>
```

Figure 3: Example of a gesticon pair: nonverbal behaviour specification for the backchannel agreement.

In this chapter we presented the SAIBA framework which is a common framework to the agent platforms which will be used to create the virtual agents which is explained in detail in Chapter 5. We also presented the standard languages i.e., FML and BML which can be used to exchange messages between modules. In the next chapter, we present an overall architecture of that could be extended to Council of Coaches project.

4 Reference Architecture Council of Coaches

We aim to reuse the work that was done in other European research projects when possible. One such project that is relevant is the EU H2020 project ARIA VALUSPA (ARIA). The aim in ARIA was to build adaptive and task-oriented dialogues in multiple languages to assist information retrieval in general. Additionally, they built a framework for information retrieval style dialogue management specification that can be used outside the ARIA-VALUSPA project for adaptive dialogue construction.

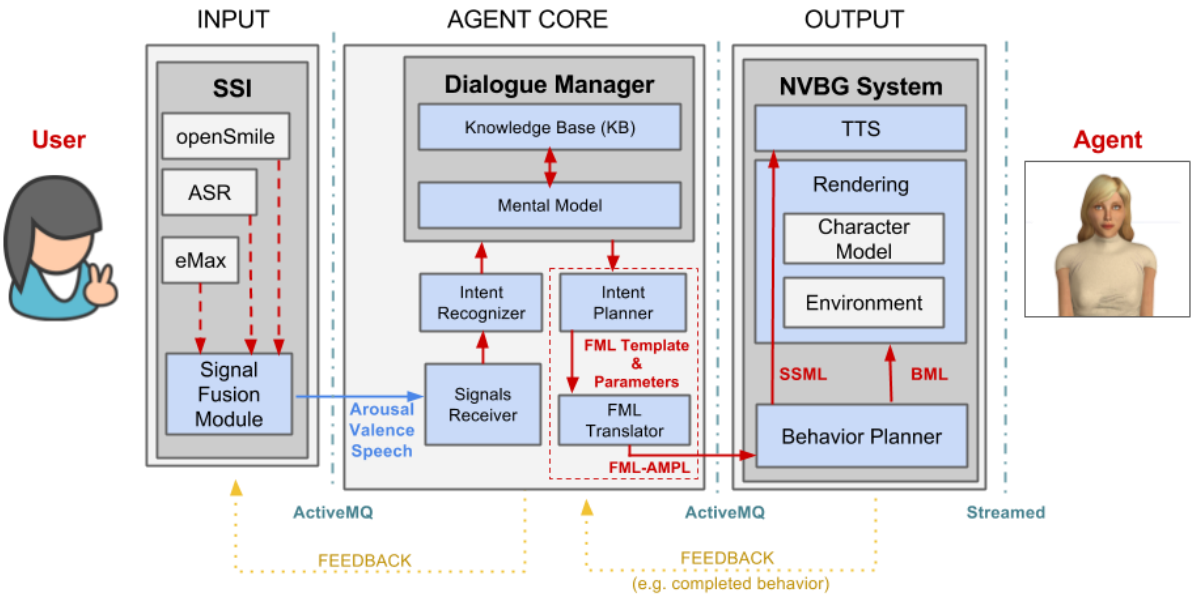

Figure 4: Technical component architecture from the EU H2020 project ARIA VALUSPA.

The architecture in the ARIA project takes the perspective of an interaction between a single user and a single agent, see Figure 4. This figure shows the technical components of the agent and their place within the interaction. Components are divided based on the task that they perform into input, core, and output. In the Input module, there are components dealing with audio-visual multimodal input detection of the user. These components detect user's speech and multimodal behaviour (e.g. facial expressions) in real-time and provide low level signals (i.e. raw detected signals such as a user's head nod) and high level signals (i.e. interpreted information, such as a user's emotional state) to the Agent Core module. The Agent Core's components keep track of the user's socio-emotional and mental states and plan the dialogue content and multimodal behaviour of the Agent. Finally, the produced speech and multimodal behaviour is handled by the Output module where a text-to-speech (TTS) component and rendering component finalise the output that needs to be sent to the Client side where it is displayed to the user on a chosen platform and device (e.g. a screen in the user's home or their mobile).

The ARIA architecture is useful for the conversational interaction component for our agents. However, our agents will have to talk to each other (multi-party) and will have to use information that is obtained outside the conversation. This is planned in the dialogue manager for the Council of Coaches and is specified in D5.1. There, it is shown how the information from the sensors is obtained, how specific coaching strategies are selected, and how the behaviour for each agent is constructed.

5 Agent Platforms

In this section, we present the two agent platforms, GRETA from UPMC and ASAP Realizer from CMC (HMI), that were developed by and currently in use by the project partners.

5.1 GRETA

The Greta platform (Pecune, Cafaro, Chollet, Philippe, & Pelachaud, 2014) is a fully SAIBA compliant system for the real-time generation and animation of ECA's verbal and nonverbal behaviour. The modular architecture of this platform supports the interconnection with external tools (e.g. SSI social signal interpretation framework (Wagner, et al., 2013), Cereproc text-to-speech engine (Aylett & Pidcock, 2007)), enhancing an agent's detection and synthesis capabilities.

Greta uses a specific implementation of FML named FML - Affective Presentation Mark-up Language (FML-APML) (Mancini & Pelachaud, 2008) which enables the expression of the degree of certainty, meta-cognitive source of information (thinking, remembering, planning), the speech act (called performative), rhetorical relations such as contradiction or cause-effect (named belief-relations), turn allocation, affect and emphasis. FML-APML relies on a taxonomy of communicative functions proposed in (Poggi I. , 2001). It defines a communicative function as a pair (meaning, signal) where meaning corresponds to the communicative value the agent wants to communicate and signal to the behaviour used to convey this meaning.

Four communicative functions are differentiated in (Poggi I. , 2001):

- **Information about speaker's beliefs**: Behaviours that provide information on the speaker's beliefs such as the degree of certainty on what they are talking about.
- **Information about speaker's intentions**: The speaker may provide information on the goal, for example, through the choice of performative or the focus of the sentence.
- **Information about speaker's affective state**: The speaker may show emotion states through particular facial expressions.
- **Metacognitive information about speaker's mental state**: The speaker may try to remember or recall information.

We define a communicative act as a pair (meaning, signal). A communicative act may be associated with different signals. That is, for a given meaning there may be several ways to communicate it. For example, the meaning 'emphasis' (of a word) may co-occur with a raised eyebrow, or a head nod, or a combination of both signals. Vice versa, the same signal may be used to convey different meanings; e.g. a raised eyebrow may be a sign of surprise, of emphasis, or even of suggestion.

5.1.1 FML-APML, Affective Presentation Mark-up Language, Function Mark-up Language

FML-APML follows the syntax of the Function Mark-up Language (Heylen, Marsella, Pelachaud, & Vilhjálmsson, 2008). Its tags are based on the 'Affective Presentation Mark-up Language' (APML) first defined in (De Carolis, Pelachaud, Poggi, & Steedman, 2004.). The tags of FML-APML are the communicative functions defined in the taxonomy of Isabella Poggi (Poggi I. , 2007). The duration of each communicative intention can be specified explicitly (in seconds) or in relation to a speech act. It is also possible to define not only the speaker's intentions but also the listener's ones. An FML-APML example is shown in Figure 5. The example is made of two parts. The top part concerns the BML part while the bottom one the FML tags.

The internal BML is used as a 'shortcut' for including speech content — which is normally an instance of verbal behaviour and thus should not be within an FML request — and synchronise it with intentions (in the FML part for example) by using temporal markers (i.e. mark). This part contains also indications about the various pitch and boundary accents. The part within the tag `<fml>` contains the communicative functions. Each function is defined with its type (e. g., performative or emotion), its specific value (e. g., greet or joy) and its starting & ending times. The time markers refer to markers indicated within the text to be said by the agent.

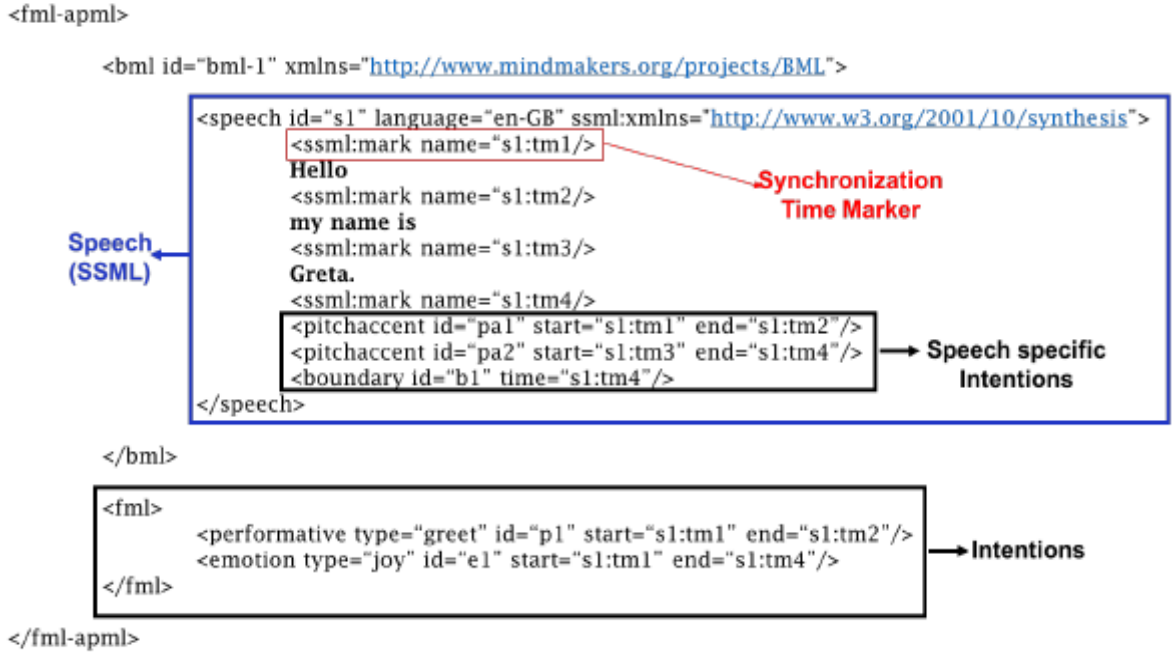

Figure 5: Example of an FML-APML request indicating the speech block specified using the SSML standard and communicative intentions.

Furthermore, the FML-APML set of tags has some interesting features regarding the timing and importance of communicative intents, the emotional state of the agent and information on the world. The timing is specified with attributes inspired by the BML recommendations (Kopp, et al., 2006) (Vilhjálmsson, et al., 2007) and makes possible absolute or relative timings of intents with symbolic labels for referencing. We will now look at each feature in detail.

5.1.2 Communicative intentions in FML-APML

Each FML-APML tag represents one communicative intention; different communicative intentions can overlap in time. We consider the following tags (taken from (Poggi I. , 2007)):

- **Certainty**: is used to specify the degree of certainty the agent intends to express; possible values: certain, uncertain, certainly not, doubt;
- **Performative**: represents the agent's performative e.g. suggest, approve, or disagree; possible values: planning, thinking, remembering;
- **Theme/Rheme**: represents the topic/comment of conversation; that is, respectively, the part of the discourse which is already known or new in the participants' conversation; Possible values: implore, order, suggest, propose, warn, approve, praise, recognise, disagree, agree, criticise, accept, advice, confirm, incite, refuse, question, ask, inform, request, announce, beg, greet;
- **Belief-relation**: corresponds to the metadiscoursive goal, i.e. the goal of stating the relationship between different parts of the discourse; Possible values: gen-spec, cause-effect, solutionhood, suggestion, modifier, justification, contrast;

- **Turn taking**: models the exchange of speaker turns; possible values: take, give;
- **Emotion**: describes the emotional state of the agent. We can define simple emotions using emotional labels (e.g. anger or sadness), but also complex emotional states like masking (i.e. the agent has a certain emotion, but it hides it by showing another, fake, one) or superposition of two emotions;
- **Emphasis**: is used to emphasise the agent's verbal or nonverbal message; possible values: low, medium, high;
- **Backchannel**: Through backchannels the listener provides information about its communicative intentions, in particular about its will and ability to continue, perceive, and understand the interaction, and its attitude towards the speaker's speech (if it believes or not, likes or not, accepts or refuses what is being said) (Allwood, Nivre, & Ahlsén, 1992);
- **World**: refers to objects of the world. For example, it can be used to indicate a point in space or the shape of an object.

5.1.3 Attributes of FML-APML tags

The attributes of FML-APML tags are:

- **Name**: the name of the tag, representing the communicative intention modelled by the tag. For example, the name performative represents a performative communicative intention;
- **ID**: a unique identifier associated to the tag; it allows one to refer to it in an unambiguous way;
- **Type**: this attribute specifies the communicative meaning of the tag. For example, a performative tag has many possible values for the type attribute e.g. suggest, propose, approve, etc. Depending on both the tag name (performative) and type (one of the above values), our Behaviour Planning module determines the nonverbal behaviours the agent has to perform;
- **Start**: starting time of the tag, in seconds. It can be absolute (time 0 corresponds to the start of the FML-APML message) or relative to another tag. It represents the point in time at which the intention specified by the tag starts to be communicated;
- **End**: duration of the tag. It can be a numeric value (in seconds) relative to the beginning of the tag or a reference to the beginning or end of another tag (or a mathematical expression involving hem). It represents the duration of the communicative intention modelled by the tag;
- **Importance**: a value between 0 and 1 which represents the probability that the communicative intention encoded by the tag is communicated through nonverbal behaviour;
- **Intensity**: certain communicative acts can be expressed with different intensities. The intensity of an emotional state is described by a value from the interval [0...1].

5.1.4 Emotion tag

Emotion has a central role in communication and ECAs should be able to communicate their emotional state in order to increase effectiveness of interaction with humans. In the FML-APML language we have introduced the emotion tag, which models the speaker's felt and expressed emotional state. The former is the emotional state the speaker is really experiencing (which can be caused by an event, a person, a situation, etc.) while the latter is the one the speaker wants to communicate to the others. These two emotional states can be completely different: for example, a person can produce a polite smile to his superior even if he is angry at him. In general, people can show their emotional state (the expressed state is the felt one), suppress it (the felt state is expressed the least) or mask it (the expressed state is different from the felt one).

In FML-APML the emotion tag allows us to specify complex emotional states. We can, for example, model situations in which our agent is feeling a particular emotional state, but simulates another emotion,

hiding the felt one. This is done by controlling the felt and expressed emotional states with the regulation attribute of the emotion tag.

The possible values of the regulation attribute are:

- **Felt**: this indicates that the tag refers to a felt emotion;
- **Fake**: this indicates that the tag refers to a fake emotion, an emotion that the agent aims at simulating;
- **Inhibit**: the emotion in the tag is felt by the agent but it aims at inhibiting it as much as possible;

Let us consider the following example:

```
<FML-APML>
  <emotion id="e1" type="anger" regulation="felt" start="0" end="3"/>
  <emotion id="e2" type="joy" regulation="fake" start="0" end="3"/>
</FML-APML>
```

The agent's real emotional state is anger (the regulation attribute of the emotion tag is set to felt) but it wants to hide it with a fake display of happiness (the regulation attribute of the emotion tag is set to fake).

5.1.5 Speech and synchronisation

FML-APML tags can be attached and synchronised to the text spoken by the agent. This is modelled by including a special tag, called speech, in the BML syntax. Within this tag, we write the text to be spoken along with synchronisation points (called time markers) which can be referred to by the other FML-APML tags. For example:

```
<FML-APML>
<bml> <speech id="s1">
<tm id="tm1"/>
what are you
<tm id="tm2"/>
doing
<tm id="tm3"/>
here
<tm id="tm4"/>
</speech>
</bml>
<fml> <emotion id="id3" type="anger" start="s1:tm2" end="s1:tm4"/>
</fml>
</FML-APML>
```

With the above code, we specify that the communicative intention of emotion starts in correspondence with the word “doing” and ends at the end of the word “here”.

In FML-APML each tag contains explicit timing data, similar to BML tags. We also maintain coherence between the two languages defined inside the SAIBA framework. So, in FML-APML we can freely define the starting and ending time of each tag, or make tags referring to each other using symbolic labels. This also allows us to specify tags that are not linked to any spoken text. That is, with FML-APML we can define the communicative intention of non-speaking agents: for example, we can represent the listener's communicative intention (e.g. the listener can have the intention to communicate that it is approving what the speaker says). FML-APML tags are used to model the agent's communicative intention. Each tag represents a communicative intention (to inform about something, to refer to a place/object/person, to express an emotional state, etc.) that lasts from a certain starting time, for a certain number of seconds.

The timing attributes start and end also allow us to model the synchronisation of the FML-APML tags. They can both assume absolute or relative values. In the first case, the attributes are numeric non-negative values, considering time 0 as the beginning of the FML-APML command. For example,

```
<emotion id="id3" type="anger" start="1.0" end="2.5"/>
```

The above line of code defines the communicative intention "*emotion=anger*" which starts after one seconds of animation and lasts 2.5 seconds.

In the second case we can specify the starting or ending time of other tags, or a mathematical operation involving them. For example:

```
<emotion id="id3" type="anger" start="s1:tm2" end="s1:tm4"/>
```

5.2 ASAP - Articulated Social Agents Platform

The Articulated Social Agents Platform (ASAP) is a state-of-the-art Behaviour Mark-up Language (BML) realizer for virtual humans. ASAP is a fully SAIBA compliant (Situation, Agent, Intention, Behaviour, Animation) platform. The ASAP platform generates real-time verbal and non-verbal behaviour, such as speech, facial expressions, and gestures, for virtual humans. The platform focusses on continuous and high responsiveness interactions. The ASAP platform can act as a back-end realizer for different embodiments, like physical robots or realistic 3D full kinematic virtual humans. The architecture of ASAP is modular, which makes it easy to extend it with other modules or engines. Different engines will handle their own parts of the behaviour specification and generate synchronised instructions for realising, such as, speech output, body gestures, postures and facial expressions. ASAP has not only been used for virtual humans, but also for controlling social robots.

The ASAP platform and the Greta platform are both SAIBA compatible platforms and therefore, they are similar from a high-level perspective. Both platforms use BML to control and define the behaviours of the agent that they render and both contain capabilities for advanced control and planning of agent behaviour such as interrupting and re-planning behaviour execution. Next, some unique options and extensions of ASAP will be presented.

5.2.1 ASAP Multi-Agent Extension

Multi-Agent BML allows agent identifiers and cross references to BML code of specific other agents. This allows the system to synchronise the behaviour of different agents or create behaviours in relation to

the other agent's (e.g. location). In Figure 6 we show three agents in our first demo of this multi-agent extension.

For example, it is possible for one agent to start talking when the other stops or to look at the agent that is talking. The planning (and potential re-planning) of these behaviours is done in real-time by the ASAP realizer.

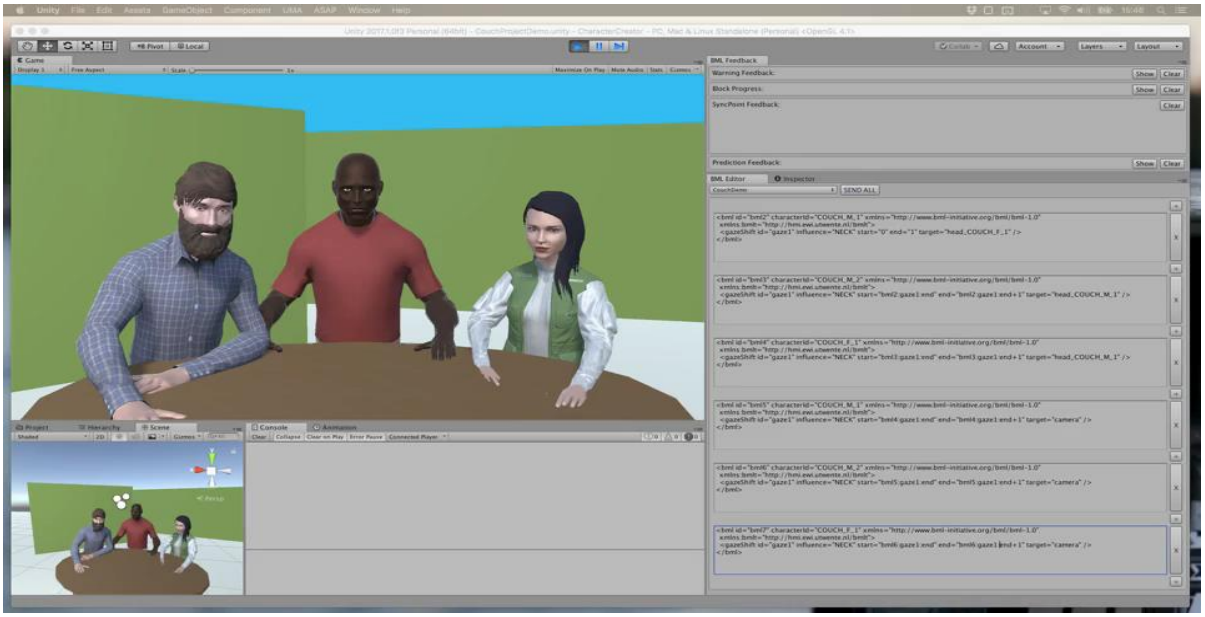

Figure 6: First demo of the Multi-agent extension of the ASAP BML realizer tool.

5.2.2 UMA: Unity Multipurpose Avatar

The Unity Multipurpose Avatar (UMA) is an open avatar creation framework plugin for Unity3D. ASAP can utilise characters created with UMA. The character in Figure 7 was created using the UMA plugin. ASAP can control these characters within the Unity3D editor (at run-time) or as exported assets in an external application. The control is achieved using a middleware connection, making it possible to control characters that run on distributed devices.

Figure 7: Agent created with UMA which can be controlled by ASAP.

ASAP controlled characters are able to interact with and react to the virtual world. For instance, they can point or look at a virtual object. To make this possible, the ASAP World Environment updater in Unity keeps ASAP updated about the location of (relevant) objects in the virtual world's coordinate system. With these coordinates, ASAP can solve the IK needed to point or gaze at a given location or object.

This is especially useful in settings where users are situated in a shared space with the agent, such as in virtual or mixed reality. Here the user's head or hands can be tracked by the VR headset or controllers and the agent can react to these objects. For more details about these capabilities see (Kolkmeier, Bruijnes, Reidsma, & Heylen, 2017)

6 Tools

In this section, we present some of the existing tools that can be used to create and manipulate multimodal, non-verbal behaviours for the two agent platforms presented in the previous chapter.

6.1 Editors

Several editors have been designed to create nonverbal signals through interfaces. Facial expressions can be defined either by manipulating each MPEG-4 parameter, Facial Animation Parameter (FAP) see Figure 8, or by manipulating action unit (AU) defined within the standard Facial Action Coding System (FACS) see Figure 9. An AU corresponds to a minimal visible facial action.

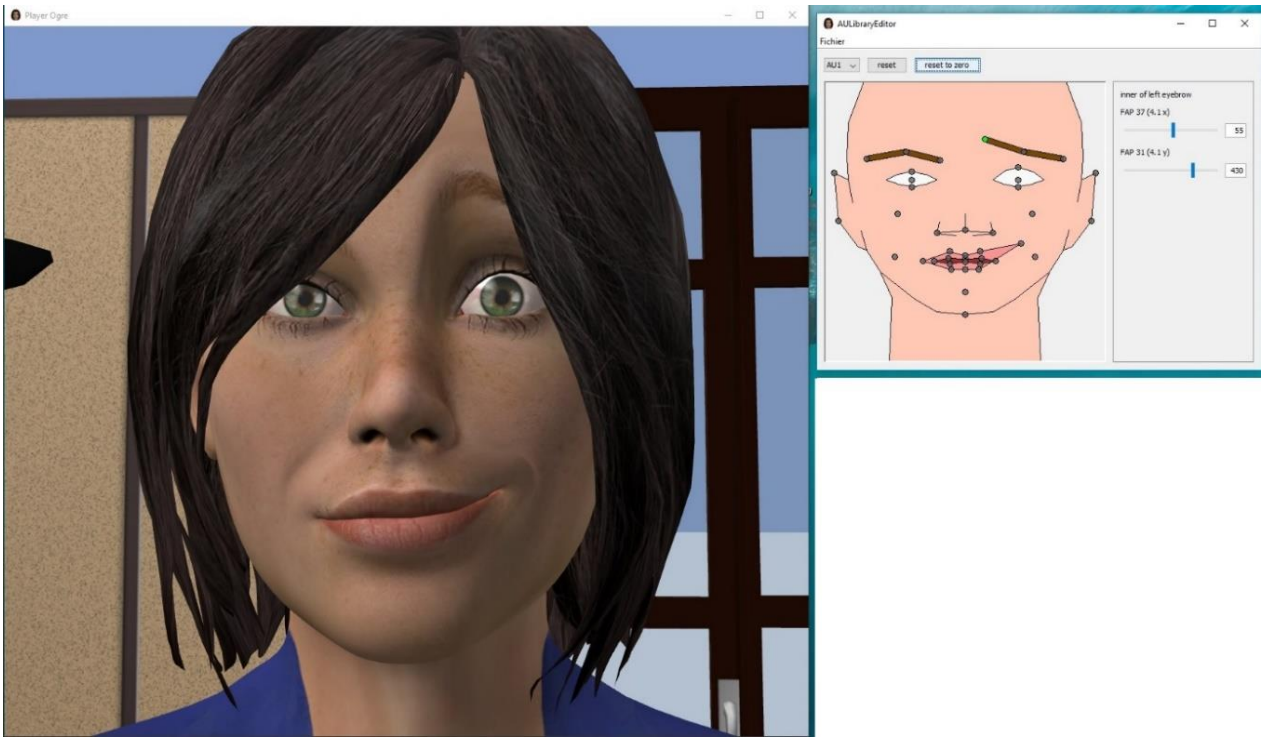

Figure 8: Facial expression created using MPEG-4.

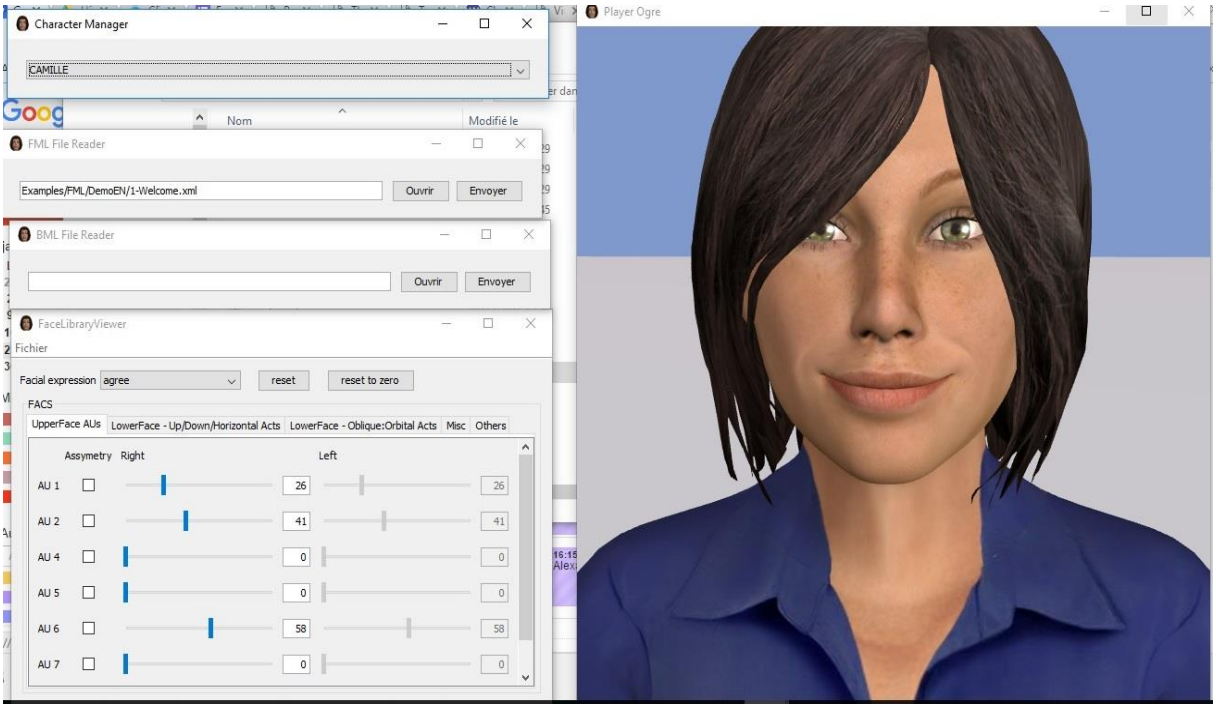

Figure 9: Facial Expression created with Action Units.

Gesture, hand shape and torso can also be defined through their own editors (see Figure 10). A gesture is characterised by its stroke phase(s). A stroke is defined by the position of the wrist in 3D space, the palm orientation, the hand shape and the movement of the wrist.

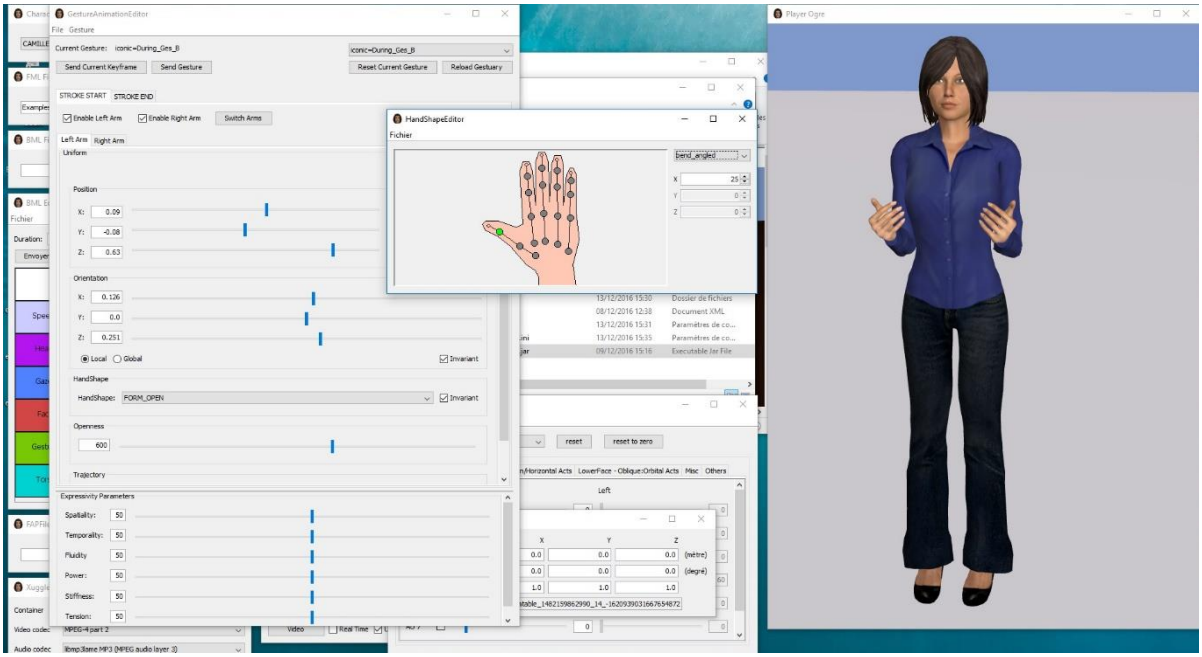

Figure 10: Editors to create gesture and hand shape.

6.2 Integration of Greta platform within Unity 3D

The Greta platform has been integrated into the Unity3D engine. The integration is realised through Thrift, a software framework for scalable cross-language services development. Unity3D and the Greta platform are connected using a TCP-IP message controller. Unity3D is the server and Greta the client which sends request when an animation must be updated.

The animation module of Unity3D had to be adapted to Greta virtual characters. This is because in Greta the animation engine is based on 3D models of the characters, animated in Ogre3D using "poses" (sub-meshes deformations); while, in Unity3D, it is based on "bones" animation; to enable a complete (body and facial expressions) animation of virtual agents we have developed a script enabling us to automatically transform the Greta-compatible character models based on meshes and ogre poses into bones based models that are compatible with Unity3D. As a result, when an input file, be it BML or FML, is sent to the Greta platform, the animation is computed as MPEG-4 parameters that can be displayed within Unity3D. Animation information is sent frame by frame in real-time which enables the Greta platform to control several agents at the same time and in real-time in Unity3D; it mixes the flexibility of Unity3D for creation of environments and scenarios and the complexity of Greta for agents control.

6.3 Creation of new characters for the Greta platform

New 3D models of virtual characters can be created with Autodesk Character Generator. A script has been developed to make the Autodesk character compatible with Greta. The following parameters are available and can be chosen at will:

- Name of the character
- Character Height
- Textures
- Clothes

For the following parameters, a specific value should be given so the 3D models are compatible with the Greta platform:

- Polygons Resolution: should be set to High
- Geometry: should be set to Triangles
- Facial Expressions: Facial bone rig
- Generate: Generic.fbx
- Skeleton Resolution: High
- Character Orientation: Y-up

7 Home and Mobile UI

Within earlier European research projects we worked on different types of user interfaces (UI) where the Greta and/or ASAP platform were integrated to generate the behaviour of virtual agents. In this chapter, we will present an overview of the different types of user interfaces on different devices.

Within the European research project Smarcos a multi-device coaching system was developed (see Figure 11) (Klaassen, et al., 2011) empowered users to make healthy lifestyle choices. This system should be able to track and interact with the users in the home, work and outdoor context. This implies that the system and the virtual coach should be interusable, which means users should be able to interact with the coach through different devices and modalities and they should recognise that they have to do so with one and the same service. The virtual coach can be presented as a virtual agent.

The ASAP realizer is used to generate the virtual agent in the Smarcos system. Coaching messages are generated in BML for ASAP realizers on the devices. Home devices (PCs and TVs) and mobile devices (Android based) in the system are "BML enabled", which means they can run the ASAP realizer and display the generated virtual human.

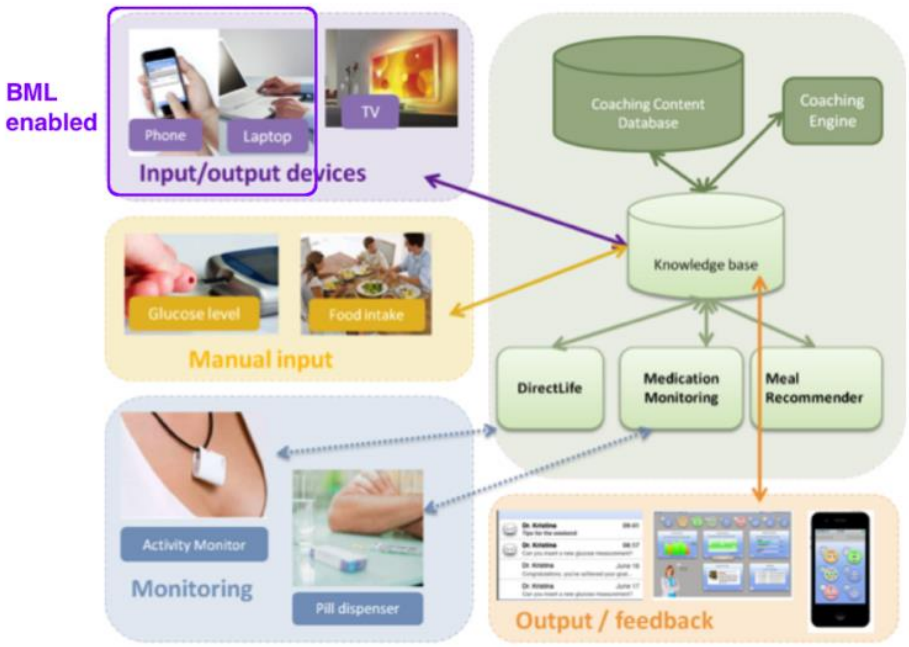

Figure 11: The Smarcos coaching system, including "BML-enabled" devices where BML driven agents are present.

A light-weight 2D PictureEngine (Klaassen, Hendrix, Reidsma, & op den Akker, 2012) embodiment was developed that allows the ASAP platform to run as a user interface of Android applications (Figure 12) and present coaching messages. On normal PCs and laptops the 3D embodiment was used to present the coaching messages (see Figure 13). Hand-overs between the different devices in the system are possible.

Figure 12: A coaching message presented by the ASAP PictureEngine agent on an Android device.

Users can continue the interaction with the coach while switching from interaction device. User information is synchronised between the devices when a hand overtakes place. Figure 14 presents the same virtual coach on two different "BML-enabled" devices.

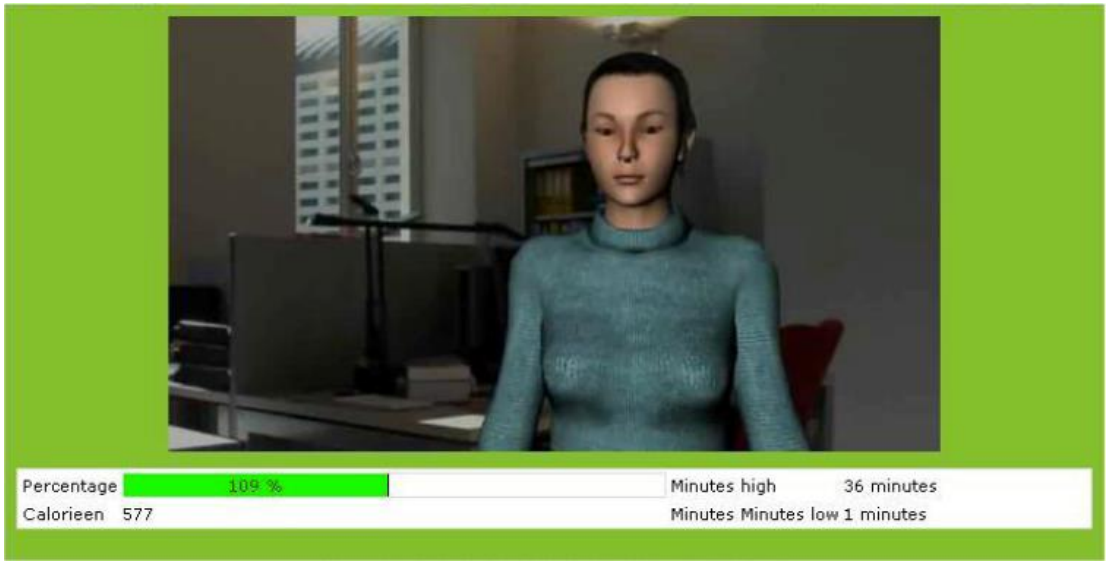

Figure 13: A coaching message presented by the ASAP 3D agent on a PC.

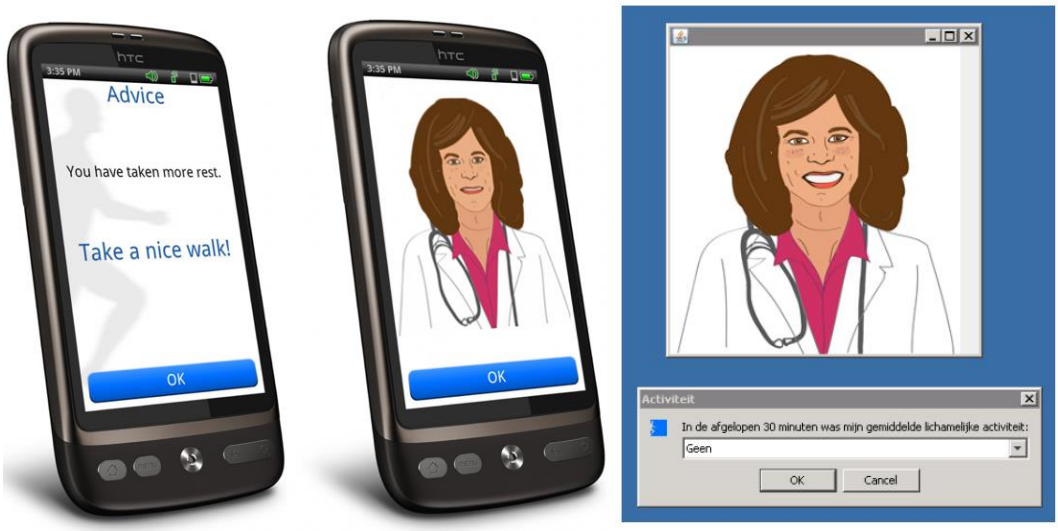

Figure 14: A coaching message presented by a virtual human and in text on 2 different devices (a mobile Android device and Windows PC).

Within the European research project PERGAMON (Klaassen, et al., 2018), we developed a gamification platform that integrates educational serious gaming and virtual coaching for children with diabetes type I. The virtual coach in the PERGAMON is not generated by a BML realizer. The virtual coach is able the send and present coaching messages (reminders, suggestions, advice and overviews) about their self-management of their diabetes and progress they made in the game. Messages from the coach were sent to a web application and to mobile (Android and iOS) devices. The virtual coach was represented by a character that fits the theme of the game. An example of a coaching message can be found in Figure 15. Messages on the web application and on the mobile devices were presented in the same way.

Figure 15: A coaching message from the PERGAMON system (in Dutch).

Finally, the Greta platform (see Section 5.1) and the ASAP platform (Kolkmeier, Bruijnes, Reidsma, & Heylen, 2017) are both integrated into the Unity3D[1] game engine. The integration of a BML realizer in the powerful animation engine of Unity3D makes it possible to combine the advantages of a BML

[1] https://unity3d.com

realisation engine (such as multimodal behaviour planning and synchronisation) and the animation engine of a game engine.

As Unity3D is specifically designed to be a cross platform game engine, it is possible to export the solutions to all kind of platforms, such as Windows, Linux, macOS, iOS, Android and Virtual Reality platforms. In this way the BML realizers are compatible with multiple devices and can facilitate developing solutions that extend across multiple platforms (e.g. such as the Council of Coaches Main App paired with a mobile "Companion App").

8 Conclusion

In conclusion, in this deliverable we presented a detailed description of the human-agent architecture, agent platforms that will be used in the Council of Coaches project along with the tools and the standard languages. The platforms and tools described in the document enable us to design Embodied Conversational Coaches with varying physical appearances, multimodal behaviours, and social attitudes. We also provide the description of an overall human-agent architecture that can be extended to a multi-agent version for this project. Further, we presented existing user interfaces that are relevant to the project.

In the next stage of this Work Package, we plan to describe the design and development of the initial user interfaces integrated with the first prototype. User interfaces will allow defining the characteristics of the virtual coaches at various levels, namely behaviours, emotion sensibility, attitude, and interactional sensibility. The focus will be on developing an easy to use, non-obtrusive user interface that allows setting up the parameters for the different characteristics. In parallel, we also aim to design the initial turn-taking mechanisms for the Embodied Conversational Coaches and develop preliminary behaviour models to ensure capturing and maintaining of user engagement.

9 Bibliography

Allwood, J., Nivre, J., & Ahlsén, E. (1992). On the semantics and pragmatics of linguistic feedback. *Journal of Semantics, 9*(1), 1-26.

Aylett, M. P., & Pidcock, C. J. (2007). The CereVoice Characterful Speech Synthesiser SDK. *Proc. of the 7th international conference on Intelligent Virtual Agents. 4722*, pp. 413-414. Springer-Verlag.

Cafaro, A., Vilhjálmsson, H., Bickmore, T., Heylen, D., & Pelachaud, C. (2014). Representing Communicative Functions in SAIBA with a Unified Function Markup Language. In T. Bickmore, S. Marsella, & C. Sidner (Eds.), *Intelligent Virtual Agents* (Vol. 8637, pp. 81-94). Springer International Publishing. Retrieved from http://dx.doi.org/10.1007/978-3-319-09767-1_11

De Carolis, B., Pelachaud, C., Poggi, I., & Steedman, M. (2004.). APML, a markup language for believable behavior generation. . *Life-Like Characters.*, 65-85.

Heylen, D., Marsella, S., Pelachaud, C., & Vilhjálmsson, H. (2008). The Next Step towards a Function Markup Language. *Proceedings of the 8th international conference on Intelligent Virtual Agents* (pp. 270-280). Berlin: Springer-Verlag.

Klaassen, R., Bul, K., op den Akker, R., van der Burg, G., Kato, P., & Di Bitonto, P. (2018). Design and Evaluation of a Pervasive Coaching and Gamification Platform for Young Diabetes Patients. *Sensors, 18*(2), 402.

Klaassen, R., Hendrix, J., Reidsma, D., & op den Akker, H. J. (2012). Elckerlyc goes mobile: enabling technology for ECAs in mobile applications. *The Sixth International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies.* XPS (Xpert Publishing Services).

Klaassen, R., Lavrysen, T., Geleijnse, G., van Halteren, A., Schwietert, H., & van der Hout, M. (2011). A personal context-aware multi-device coaching service that supports a healthy lifestyle. *In Proceedings of the 25th BCS Conference on Human-Computer Interaction* (pp. 443-448). British Computer Society.

Kolkmeier, J., Bruijnes, M., Reidsma, D., & Heylen, D. (2017). An ASAP Realizer-Unity3D Bridge for Virtual and Mixed Reality Applications. *Intelligent Virtual Agents.* Stockholm.

Kopp, S., Krenn, B., Marsella, S. C., Marshall, A., Pelachaud, C., Pirker, H., . . . Vilhjálmsson, H. (2006, 8). Towards a Common Framework for Multimodal Generation: The Behavior Markup Language. *Proceedings of the Intelligent Virtual Humans Conference.* Marina del Rey, CA.

Kronlid, F. (2006). Turn taking for artificial conversational agents. *Cooperative Information Agents, Springer Berlin / Heidelberg. volume 4149*, 81-95.

Mancini, M., & Pelachaud, C. (2008). The FML-APML language. *Why Conversational Agents do what they do. Workshop on Functional Representations for Generating Conversational Agents Behavior at AAMAS.* Estoril.

Op den Akker, H. J., & Bruijnes, M. (2012). *Computational models of social and emotional turn-taking for embodied conversational agents.* Enschede, the Netherlands: University of Twente.

Pecune, F., Cafaro, A., Chollet, M., Philippe, P., & Pelachaud, C. (2014). Suggestions for Extending SAIBA with the VIB Platform. *Workshop on Architectures and Standards for IVAs, held at the '14th International Conference on Intelligent Virtual Agents (IVA 2014)'* (pp. 16–20). Boston: Bielefeld eCollections.

Poggi, I. (2001). Mind markers. *The Semantics and Pragmatics of Everyday Gestures.* .

Poggi, I. (2007). *Mind, Hands, Face and Body. A Goal and Belief View of Multimodal Communication* (Vols. Körper, Zeichen, Kultur). Weidler Verlag.

Sidner, C., & Rich, C. (2012). Procedural Dialogue Authoring with Hierarchical Task Networks and Dialogue Trees. *Proc. of the 12th International Conference on Intelligent Virtual Agents.*

Vilhjálmsson, H. H. (2009). Representing Communicative Function and Behavior in Multimodal Communication. In A. Esposito, A. Hussain, M. Marinaro, & R. Martone (Eds.), *Multimodal Signals: Cognitive and Algorithmic Issues* (Vol. 5398, pp. 47-59). Springer Berlin Heidelberg.

Vilhjálmsson, H. H., Cantelmo, N., Cassell, J., E. Chafai, N., Kipp, M., Kopp, S., . . . Werf, R. J. (2007). The Behavior Markup Language: Recent Developments and Challenges. *Proceedings of the 7th international conference on Intelligent Virtual Agents* (pp. 99-111). Berlin: Springer-Verlag.

Wagner, J., Lingenfelser, F., Baur, T., Damian, I., Kistler, F., & André, E. (2013). The social signal interpretation (SSI) framework: multimodal signal processing and recognition in real-time. *Proceedings of the 21st ACM international conference on Multimedia* (pp. 831-834). New York, NY, USA: ACM.