SEMAINE

THE SENSITIVE AGENT PROJECT

D1d

Final SAL system

Date: 24 September 2010

Dissemination level: Public

ICT project contract no.	211486
Project title	**SEMAINE** **Sustained Emotionally coloured Machine-human Interaction using Nonverbal Expression**
Contractual date of delivery	*31 August 2010*
Actual date of delivery	*24 September 2010*
Deliverable number	D1d
Deliverable title	Final SAL system
Type	Demonstrator
Number of pages	27
WP contributing to the deliverable	WP 1
Responsible for task	Marc Schröder (marc.schroeder@dfki.de)
Author(s)	Marc Schröder, Elisabetta Bevacqua, Florian Eyben, Hatice Gunes, Mark ter Maat, Sathish Pammi, Etienne de Sevin, Michel Valstar, Martin Wöllmer
EC Project Officer	Philippe Gelin

Table of Contents

1 Executive Summary

Sensitive Artificial Listeners (SAL) are virtual dialogue partners who, despite their very limited verbal understanding, intend to engage the user in a conversation by paying attention to the user's emotions and non-verbal expressions. The SAL characters have their own emotionally defined personality, and attempt to drag the user towards their dominant emotion, through a combination of verbal and non-verbal expression.

This report is part of the series of reports describing the implementation of SAL in system SEMAINE-3.0. The software described, and the full set of reports, can be downloaded from http://semaine.opendfki.de/wiki/SEMAINE-3.0/.

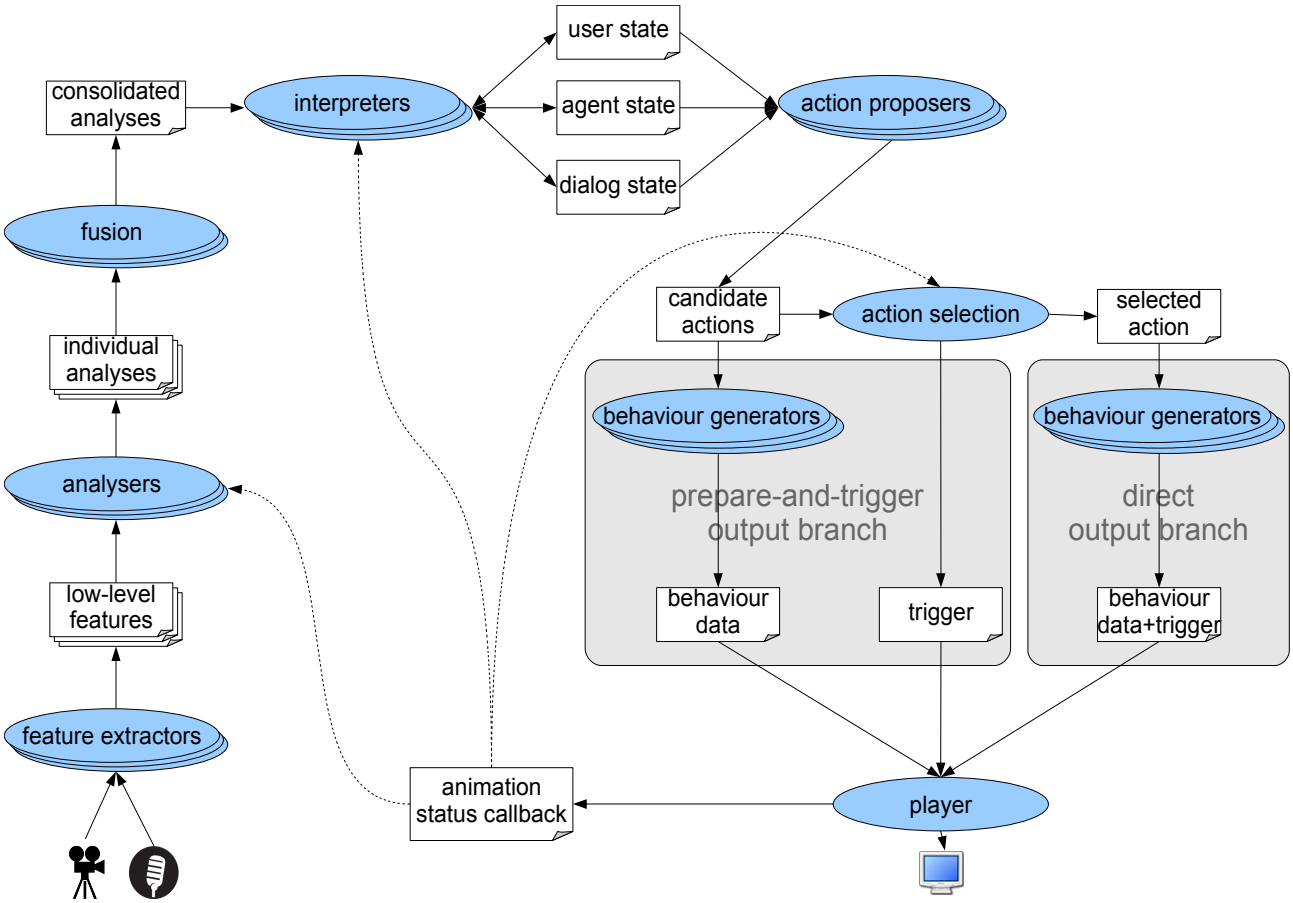
This report describes the overall structure of the SEMAINE-3.0 system, in terms of components that communicate over Topics in various representation formats.

2 Architecture of the SEMAINE-3.0 system

The architecture of the SEMAINE-3.0 system has been extended and cleaned up compared to the previous version SEMAINE-2.0. New components were added; the structure of the architecture was homogenised on the input side; and the output side was extended with a prepare-and-trigger mechanism to speed up system responses (see SEMAINE report D5b for details).

Figure 1 shows a schematic Message Flow Graph of the full system. Comparing it with the previous version of this conceptual architecture graph (Schröder, 2010), three main changes can be identified:

- the input side of the architecture now includes fusion components between the analysers and the interpreters;
- the output side of the architecture distinguishes between two output branches, a direct one and a prepare-and-trigger branch;
- the information about the current state of the player output is communicated via a dedicated animation status callback communication channel.



The following subsections describe the three main parts of the system in more detail: analysis of user behaviour, dialogue management, and generation of agent behaviour.

2.1 Analysis of user behaviour

User behaviour is first represented in terms of low-level audio and video features, then in terms of individual analysis results, and then fused together before the dialog model's "current best guess" of the user state is updated.

More details about the analysis of user behaviour can be found in SEMAINE deliverable reports D2b and D3c.

2.1.1 Feature extractors

Feature extractors are modality-specific. They produce feature vectors as key-value pairs, which are typically produced at a fixed frame rate (e.g., every 10 ms for audio features, and for every video frame for video features). The following Topics are currently used.

Topic	Description
semaine.data.analysis.features.voice	• F0frequency [0,600] (fundamental frequency in Hz) • voiceProb [0,1] (probability that the current frame is harmonic) • RMSenergy [0, 1] (energy of the signal frame) • LOGenergy [-100,0] (energy of the signal frame, in dB) (more upon request...!)
semaine.data.analysis.features.video.facedetection	• xPositionTopLeft [0,xCameraResolution] (top left corner of the bounding box of the face detected) • yPositionTopLeft [0, yCameraResolution] (top left corner of the bounding box of the face detected) • width [0,xCameraResolution] (width of the bounding box of the face detected) • height [0,yCameraResolution] (height of the bounding box of the face detected) (all 0 if no face detected)
semaine.data.analysis.features.video.facialpoints	xRightOuterEyeCorner yRightOuterEyeCorner xLeftOuterEyeCorner yLeftOuterEyeCorner xRightInnerEyeCorner yRightInnerEyeCorner

	xLeftInnerEyeCorner yLeftInnerEyeCorner xRightInnerBrowCorner yRightInnerBrowCorner xLeftInnerBrowCorner yLeftInnerBrowCorner xRightOuterBrowCorner yRightOuterBrowCorner xLeftOuterBrowCorner yLeftOuterBrowCorner xRightUpperEyelid yRightUpperEyelid xLeftUpperEyelid yLeftUpperEyelid xRightLowerEyelid yRightLowerEyelid xLeftLowerEyelid yLeftLowerEyelid xRightNostril yRightNostril xLeftNostril yLeftNostril xRightMouthCorner yRightMouthCorner xLeftMouthCorner yLeftMouthCorner xUpperLip yUpperLip xLowerLip yLowerLip xChin yChin xNose yNose xRightPupil yRightPupil xLeftPupil yLeftPupil
semaine.data.analysis.features.video.2dheadmotion	• motionDirection [-π, π] (angle of the motion) • motionMagnitudeNormalised [0, large number] (pixels per frame) • motionX [-large number, large number] (pixels per frame) • motionY [-large number, large number] (pixels per frame)

semaine.data.analysis.features.video.faceappears	
semaine.data.analysis.features.video.geomfacs	
semaine.data.analysis.features.video.lbpfacs	

2.1.2 Analysers

Analysers produce three types of information: the verbal content recognised; the user's non-verbal behaviour; and the user's emotions. Analysers represent their output as EMMA messages (Johnston et al., 2009), i.e. the specific analysis output is accompanied by a time stamp and a confidence.

Topic	Description
semaine.data.state.user.emma.words	verbal content recognised
semaine.data.state.user.emma.nonverbal.voice	voice analysis includes: • voice activity detection • stylised pitch movements • non-verbal vocalizations
semaine.data.state.user.emma.nonverbal.face	face analysis includes: • face presence • facial expression encoded in Action Units
semaine.data.state.user.emma.nonverbal.head	head gestures such as nods, shakes etc.
semaine.data.state.user.emma.emotion.voice	emotion as recognised from the voice
semaine.data.state.user.emma.emotion.face	emotion as recognised from the face
semaine.data.state.user.emma.emotion.head	emotion as recognised from head movements

Verbal content is represented directly in EMMA. For example:

```
<emma:emma version="1.0"
  xmlns:emma="http://www.w3.org/2003/04/emma">
  <emma:sequence emma:offset-to-start=”12345” emma:duration=”110”>
    <emma:interpretation
      emma:offset-to-start="12345"
      emma:tokens="bla"
      emma:confidence="0.3"/>
    <emma:interpretation
      emma:offset-to-start="12390"
      emma:tokens="bloo"
      emma:confidence="0.4"/>
  </emma:sequence>
</emma:emma>
```

The output of the **voice activity detection** (VAD) / the speaking detector looks like this. It needs no confidence. The Speaking Analyser (part of the TumFeatureExtractor) outputs messages when the

user starts or stops speaking. These messages are low-level messages, created directly from the VAD output, smoothed only over 3 frames. Thus, some thresholds must be applied in other components to reliably detect continuous segments where the user is speaking and avoid false alarms.

```
<emma:emma version="1.0" xmlns:emma="http://www.w3.org/2003/04/emma">
    <emma:interpretation  emma:offset-to-start="12345" emma:confidence="0.3">

        <semaine:speaking xmlns:semaine="http://www.semaine-project.eu/semaineml"
statusChange="start"/>

    </emma:interpretation>
</emma:emma>
```

Possible values for /emma:emma/emma:interpretation/semaine:speaking/@statusChange : start, stop

Stylised pitch movements are represented as follows:

```
<emma:emma version="1.0" xmlns:emma="http://www.w3.org/2003/04/emma">
    <emma:interpretation  emma:offset-to-start="12345" emma:duration="444"
emma:confidence="0.3">

        <semaine:pitch xmlns:semaine="http://www.semaine-project.eu/semaineml"
direction="rise"/>

    </emma:interpretation>
</emma:emma>
```

Possible values for /emma:emma/emma:interpretation/semaine:pitch/@direction : rise, fall, rise-fall, fall-rise, high, mid, low

User gender is encoded as shown here:

```
<emma:emma version="1.0" xmlns:emma="http://www.w3.org/2003/04/emma">
    <emma:interpretation  emma:offset-to-start="12345" emma:confidence="0.3">

      <semaine:gender name="female" xmlns:semaine="http://www.semaine-
project.eu/semaineml"/>

    </emma:interpretation>
</emma:emma>
```

Possible values of /emma:emma/emma:interpretation/semaine:gender/@name : male, female, unknown

Non-verbal user vocalizations such as laugh, sigh etc. are encoded as shown in the following example:

```
<emma:emma version="1.0"    xmlns:emma="http://www.w3.org/2003/04/emma">
    <emma:interpretation  emma:offset-to-start="12345" emma:confidence="0.3">

        <semaine:vocalization xmlns:semaine="http://www.semaine-project.eu/semaineml"
name="(laughter)"/>

    </emma:interpretation>
</emma:emma>
```

Values of /emma:emma/emma:interpretation/semaine:vocalization/@name are currently "(laughter)", "(sigh)" and (breath)".

Whether there is a **face present** is encoded as follows:

```
<emma:emma version="1.0" xmlns:emma="http://www.w3.org/2003/04/emma">
    <emma:interpretation  emma:offset-to-start="12345" emma:confidence="0.3">

        <semaine:face-present xmlns:semaine="http://www.semaine-project.eu/semaineml"
statusChange="start"/>

    </emma:interpretation>
</emma:emma>
```

Possible values for /emma:emma/emma:interpretation/semaine:face-present/@statusChange : start, stop

Any **action units** recognised from the user's face are encoded such that a separate confidence can be given for each action unit. Example:

```
<emma:emma version="1.0" xmlns:emma="http://www.w3.org/2003/04/emma">
  <emma:group>
    <emma:interpretation  emma:offset-to-start="12345" emma:confidence="0.3">
        <bml:bml xmlns:bml="http://www.mindmakers.org/projects/BML">
            <bml:face au="1"/>
        </bml:bml>
    </emma:interpretation>
    <emma:interpretation  emma:offset-to-start="12345" emma:confidence="0.4">
        <bml:bml xmlns:bml="http://www.mindmakers.org/projects/BML">
            <bml:face au="2"/>
        </bml:bml>
    </emma:interpretation>
    <emma:interpretation  emma:offset-to-start="12345" emma:confidence="0.2">
        <bml:bml xmlns:bml="http://www.mindmakers.org/projects/BML">
            <bml:face au="4"/>
        </bml:bml>
    </emma:interpretation>
  </emma:group>
</emma:emma>
```

Head gestures such as nods or shakes are represented as follows:

```
<emma:emma version="1.0"    xmlns:emma="http://www.w3.org/2003/04/emma">
    <emma:interpretation  emma:offset-to-start="12345" emma:duration="444"
emma:confidence="0.3">

      <bml:bml xmlns:bml="http://www.mindmakers.org/projects/BML">
          <bml:head type="NOD" start="12.345" end="12.789"/>
      </bml:bml>

    </emma:interpretation>
</emma:emma>
```

Possible values for /emma:emma/emma:interpretation/bml:bml/bml:head/@type: NOD, SHAKE, TILT-LEFT, TILT-RIGHT, APPROACH, RETRACT. Left and right are defined subject centred (i.e. left is left for the user).

For all **emotion** messages, the information is encoded using the latest draft of the EmotionML standard (Schröder et al., 2010) as the payload of an EMMA container. For example:

```
<emma:emma xmlns:emma="http://www.w3.org/2003/04/emma" version="1.0">
  <emma:interpretation>
    <emo:emotion xmlns:emo="http://www.w3.org/2009/10/emotionml"
             dimension-set="http://www.example.com/emotion/dimension/FSRE.xml">
      <emo:dimension confidence="0.905837" name="arousal" value="0.59999996"/>
      <emo:dimension confidence="0.97505563" name="valence" value="0.42333332"/>
      <emo:dimension confidence="0.9875278" name="unpredictability" value="0.29333335"/>
      <emo:dimension confidence="0.96318215" name="potency" value="0.31333336"/>
      <intensity confidence="0.94343144" value="0.04"/>
    </emo:emotion>
   </emma:interpretation>
</emma:emma>
```

2.1.3 Fusion components

All non-verbal analyses are combined by a NonverbalFusion component; all emotion analyses are combined by an EmotionFusion component, which computes the fused positions on emotion dimensions as a sum of individual positions weighted by the respective confidences.

Topic	Description
semaine.data.state.user.emma.nonverbal	consolidated non-verbal behaviour from all available modalities
semaine.data.state.user.emma.emotion	consolidated and fused emotion analysis based on information from all available modalities

2.2 Dialogue management

The dialogue management is performed by interpreters and action proposers operating on "state" information, i.e. the system's "current best guess" regarding the state of the user, the agent itself and their dialogue.

More information about the dialogue management can be found in SEMAINE deliverable report D4b.

2.2.1 Interpreters

Interpreters take both the analysis results and the existing state information into account when making various interpretations, leading to various state updates. Analyses are thresholded by confidence – only analyses with a sufficiently high confidence lead to a state update.

- EmotionInterpreter updates the user's emotion state based on the fused emotion analyses;
- NonVerbalInterpreter updates the user's nonverbal state based on the fused non-verbal analyses;
- UtteranceInterpreter updates the user state with respect to the words spoken by the user, taking the current dialogue state into account;

- TurnTakingInterpreter takes decisions on the agent's intention to take the turn, based on current user, dialogue and agent state, and updates dialog and agent state accordingly;
- AgentMentalStateInterpreter updates the agent's mental state based on user behaviour.

State information is kept in three Topics:

Topic	Description
semaine.data.state.user.behaviour	all “current best guess” information about the user
semaine.data.state.agent	the current state of the agent
semaine.data.state.dialog	all “current best guess” information regarding the state of the dialog

The state information is accessed via a short name and encoded in XML according to a stateinfo.config file. The latest version of the stateinfo.config file is enclosed as Appendix A.

2.2.2 Action proposers

There are currently two action proposer components.

- UtteranceActionProposer is simultaneously proposer and action selection for verbal utterances. It selects suitable utterances from the available set of utterances for the current character, based on the current state information, and triggers the most suitable one when the agent has a turn-taking intention. This component manages the two output cues, see below.
- ListenerIntentPlanner proposes possible timing and meaning of reactive listener backchannels, as well as non-verbal mimicry by the agent while being a listener. It uses user and agent state information to trigger and select both reactive (meaning-based) and mimicry (behaviour-based) backchannels.

A dedicated (listener) ActionSelection component makes sure that the amount of listener actions stays moderate, and in particular holds back any backchannel intentions while the agent is currently producing a verbal utterance.

Candidate actions are produced to the following Topics.

Topic	Description
semaine.data.action.candidate.function	candidate actions described in terms of the function or meaning of what is to be expressed
semaine.data.action.candidate.behaviour	candidate actions described in terms of concrete behaviours
semaine.data.action.selected.function	selected actions described in terms of the function or meaning of what is to be expressed
semaine.data.action.selected.behaviour	selected actions described in terms of concrete behaviours

The format of candidate and selected actions is identical. Actions in Topics *.function are encoded in FML. This includes verbal utterances like the following:

```
<?xml version="1.0" encoding="UTF-8"?><fml-apml version="0.1">
   <bml:bml xmlns:bml="http://www.mindmakers.org/projects/BML" id="bml_uap_3">
      <bml:speech id="speech_uap_3" language="en-GB" text="Is that so? Tell me about it."
voice="activemary">
         <ssml:mark xmlns:ssml="http://www.w3.org/2001/10/synthesis"
name="speech_uap_3:tm1"/>Is<ssml:mark xmlns:ssml="http://www.w3.org/2001/10/synthesis"
name="speech_uap_3:tm2"/>that<ssml:mark xmlns:ssml="http://www.w3.org/2001/10/synthesis"
name="speech_uap_3:tm3"/>so?<ssml:mark xmlns:ssml="http://www.w3.org/2001/10/synthesis"
name="speech_uap_3:tm4"/>Tell<ssml:mark xmlns:ssml="http://www.w3.org/2001/10/synthesis"
name="speech_uap_3:tm5"/>me<ssml:mark xmlns:ssml="http://www.w3.org/2001/10/synthesis"
name="speech_uap_3:tm6"/>about<ssml:mark xmlns:ssml="http://www.w3.org/2001/10/synthesis"
name="speech_uap_3:tm7"/>it.<ssml:mark xmlns:ssml="http://www.w3.org/2001/10/synthesis"
name="speech_uap_3:tm8"/>
      </bml:speech>
   </bml:bml>
   <fml:fml xmlns:fml="http://www.mindmakers.org/fml" id="fml_uap_3">
      <fml:performative end="speech_uap_3:tm4" id="tag1" importance="1"
start="speech_uap_3:tm2" type="like"/>
      <fml:emotion end="speech_uap_3:tm4" id="tag2" importance="1"
start="speech_uap_3:tm2" type="small-surprise"/>
      <fml:performative end="speech_uap_3:tm6" id="tag3" importance="1"
start="speech_uap_3:tm4" type="agree"/>
   </fml:fml>
</fml-apml>
```

Reactive backchannels are also encoded in FML:

```
<fml-apml>
  <fml xmlns="http://www.mindmakers.org/fml" id="fml1">
    <backchannel end="1.8" id="b0" importance="1.0" start="0.0" type="understanding"/>
    <backchannel end="1.8" id="b1" importance="1.0" start="0.0" type="disagreement"/>
    <backchannel end="1.8" id="b2" importance="1.0" start="0.0" type="belief"/>
  </fml>
</fml-apml>
```

Mimicry backchannels are encoded in BML:

```
<bml xmlns="http://www.mindmakers.org/projects/BML">
  <head end="1.8" id="s1" start="0.0" stroke="1.0">
    <description level="1" type="gretabml">
      <reference>head=head_shake</reference>
    </description>
  </head>
</bml>
```

2.3 Generation of agent behaviour

Any agent actions must be generated in terms of low-level player data before they can be rendered. In addition to the direct generation branch, the current architecture now also supports a prepare-and-trigger branch.

More information about the generation of agent can be found in SEMAINE deliverable report D5b.

2.3.1 Direct branch

In the direct branch, a selected action is converted into player data using the following intermediate steps.

- SpeechPreprocessor computes the accented syllables and any phrase boundaries as anchors to which any gestural behaviour can be attached. It reads from Topicvs semaine.data.ac-

tion.selected.function and semaine.data.action.selected.behaviour, and writes its results to semaine.data.action.selected.speechpreprocessed. Whereas conceptually this processing step could work with purely symbolic specification of prosody-based anchor points, the current implementation of the behaviour planner component requires absolute timing information. For this reason the output of the SpeechPreprocessor already contains the detailed timing information.

- BehaviourPlanner determines suitable behaviour elements based on the intended function/meaning of the action. It uses character-specific behaviour lexicons to map FML to BML. It reads from semaine.data.action.selected.speechpreprocessed and writes to semaine.data.synthesis.plan.
- SpeechBMLRealiser carries out the actual speech synthesis, i.e. the generation of audio data. It reads from semaine.data.synthesis.plan; it writes a BML message including the speech timings to semaine.data.synthesis.plan.speechtimings, and the binary audio data including a file header to semaine.data.synthesis.lowlevel.audio.
- BehaviorRealizer reads from semaine.data.synthesis.plan.speechtimings and semaine.data.action.selected.behaviour, and produces the low-level video features in Topics semaine.data.synthesis.lowlevel.video.FAP and semaine.data.synthesis.lowlevel.video.BAP. In addition, it sends two types of information to Topic semaine.data.synthesis.lowlevel.command: (1) the information which modalities form part of a given animation as identified by a unique content ID; and (2) the trigger commands needed to start playing back the animation.
- PlayerOgre is the audiovisual player component. It reads the lowlevel player data from topics semaine.data.synthesis.lowlevel.audio, semaine.data.synthesis.lowlevel.video.FAP and semaine.data.synthesis.lowlevel.video.BAP. A unique content ID is used to match the various parts of a multimodal animation to be rendered. Two types of information are received via the Topic semaine.data.synthesis.lowlevel.command: the information which modalities are expected to be part of a given animation / content ID; and the trigger to start playing the animation. The Player sends callback messages to the topic semaine.callback.output.Animation, to inform about the preparation or playback state of the various animations it receives.

An example of a callback message is the following:

```
<callback xmlns="http://www.semaine-project.eu/semaineml">
  <event data="Animation" id="fml_lip_70" time="1116220" type="start"/>
</callback>
```

Possible event types are “ready”, “start”, “end”, as well as “stopped” and “deleted”.

2.3.2 Prepare-and-trigger branch

The prepare-and-trigger branch replicates the processing pipeline of the direct branch, but using different Topics:

- QueuingSpeechPreprocessor reads from semaine.data.action.prepare.function and semaine.data.action.prepare.behaviour, and writes to semaine.data.action.prepare.speechpreprocessed;
- BehaviorPlannerPrep reads from semaine.data.action.prepare.speechpreprocessed and writes to semaine.data.synthesis.prepare;

- QueuingSpeechBMLRealiser reads from semaine.data.synthesis.prepare and writes to semaine.data.synthesis.prepare.speechtimings and semaine.data.synthesis.lowlevel.audio;
- BehaviorRealizerPrep reads from semaine.data.synthesis.prepare.speechtimings and semaine.data.action.prepare.behaviour and writes to semaine.data.synthesis.lowlevel.video.FAP, semaine.data.synthesis.lowlevel.video.BAP and semaine.data.synthesis.lowlevel.command.

The difference to the direct branch is at the two ends of the processing pipeline. At the input end, the UtteranceActionProposer feeds into semaine.data.prepare.function candidate utterances that the current character may perform in the near future. At the output end, the BehaviorRealizerPrep sends the content-level description to the player but it does not send the trigger commands. Instead, when the player has received all necessary parts of a given animation, it sends a "ready" callback message which is then registered by the UtteranceActionProposer. When its utterance selection algorithm determines that the selected utterance already exists in prepared form in the player, all it needs to do is send a trigger command directly from UtteranceActionPoposer to semaine.data.synthesis.lowlevel.command, which then starts the playback of the prepared animation without any further delay. If no prepared version of the selected utterance is available, e.g. because it was unexpected that this utterance was selected, or because the preparation has not completed yet, the utterance is generated using the direct branch.

The prepare-and-trigger branch is used only for full utterances. Listener actions are so short and fast to generate that they always use the direct branch.

Since both branches are technically completely independent, this architecture scales well to multiple computers: it is easy to run the direct branch on one computer and the prepare-and-trigger branch on a different computer if they jointly would over-stretch the CPU resources of a single PC.

2.4 Protocol for the Player in SEMAINE

Any player component in SEMAINE must follow the following protocol so that it supports ahead-of-time preparation of possible utterances. The player must keep a collection of "Animations" which can be played by a "playCommand".

This protocol is currently implemented by two players: The audio-visual Windows native Player-Ogre using the Greta agent, and the speech-only player in Java class eu.semaine.components.-mary.QueuingAudioPlayer.

2.4.1 Data flow

Low-level player data is sent to the player via the Topics

```
semaine.data.synthesis.lowlevel.*
```

currently

```
semaine.data.synthesis.lowlevel.audio
semaine.data.synthesis.lowlevel.video.FAP
semaine.data.synthesis.lowlevel.video.BAP
```

Incoming messages have the following properties:

- a message type specific to the payload format (currently: BytesMessage for audio, TextMessage for FAP and BAP).
- a data type (obtained by message.getDatatype()) identifying the type of message (current values are "AUDIO", "FAP" and "BAP").
- a content ID and a content creation time (obtained by message.getContentID() and message.getContentCreationTime()) which are used to assemble an Animation, to match data and command messages, and to identify the content item in callback and log messages.

The idea is that a unit of player data (an "Animation") is assembled in the player from the individual data items that are coming in (currently, AUDIO, FAP and BAP). Certain data types are optional (currently: AUDIO). A message can either contain the complete data of the given type (currently the case for AUDIO) or it can contain a chunk of data (currently the case for FAP and BAP). A chunk contains information about its position in the Animation; it can be dynamically added even if the Animation is already playing.

2.4.2 Command messages

There are two types of command messages: messages with data types `dataInfo` and `playCommand`.

- Data info commands: For a given content ID, define the data types that must be present in the Animation: HASAUDIO, HASFAP and HASBAP. Each can be 0 for "not needed" or 1 for "needed".
- Player commands: For a given content ID, define the playback conditions. This includes the following aspects:
  - STARTAT: when to start the playback of the Animation (in milliseconds from the moment when the Animation becomes ready);

- PRIORITY: the priority of the Animation in case of competing Animations;
- LIFETIME: the lifetime of the Animation, counting from the moment when the animation becomes ready. When the lifetime is exceeded and the animation has not started playing, it will be marked as "dead" and removed.

Commands are sent to topic

```
semaine.data.synthesis.lowlevel.command
```

and have the data type `dataInfo` for data info commands and `playCommand` for player start trigger commands.

For every content ID, a playCommand is required in order to play that animation. Without a matching playCommand, an animation will never be played.

A command has the following format:

- its content ID is **identical** to the content ID of the Animation for which it defines playback conditions;
- message format is **TextMessage**; the text consists of space-separated key-value pairs, one pair per line, where the keys are strings and the values are floating point numbers.

The following features are used:

- for playCommand:
  - STARTAT (0 means start at the moment when all required parts are present, a positive number means milliseconds after that condition is met)
  - LIFETIME (in milliseconds from the moment the animation is triggered; -1 means it will never expire)
  - PRIORITY (a value between 0 and 1, where 0 is the lowest and 1 the highest possible priority)
- for dataInfo:
  - HASAUDIO (a binary feature, 0 means the Animation does not have audio, 1 means the Animation has audio data)
  - HASFAP (a binary feature, 0 means the Animation does not have FAP data, 1 means the Animation has FAP data)
  - HASBAP (a binary feature, 0 means the Animation does not have BAP data, 1 means the Animation has BAP data)

Every player command must contain all features of its respective type.

2.4.3 Callback messages

Event-based callback messages are sent when certain conditions are met for a given Animation. The messages go to Topic

```
semaine.callback.output.player
```

and have the following format:

```
<callback xmlns="http://www.semaine-project.eu/semaineml">
   <event id="CONTENT_ID" time="META_TIME" type="EVENT_TYPE"/>
</callback>
```

where content ID and meta time are like before, and type is one of the following:

- `ready` means the Animation has received all required data, so it is ready for playing back. This event is triggered independently of the question whether a command has been received or not.
- `deleted` means the Animation was removed before it started playing, e.g. because it has exceeded its lifetime in the output queue.
- `start` means the Animation has started playing.
- `stopped` means the Animation was stopped while playing but before it was finished, e.g. because a request to change character was received.
- `end` means the Animation has finished playing.

2.4.4 Error conditions

The content ID must be unique for the lifetime of a system. This leads to the following error conditions.

It is an error condition...

- if a data chunk is received for an Animation that has already been discarded (because it finished playing, or exceeded its lifetime in the queue);
- if data is received for a data type that does not form chunks;
- if a playCommand is received for a content ID that has been started already, or that is already discarded;

An error condition should be reported as a WARN log message, and otherwise ignored.

It is **not** an error condition...

- if a second playCommand is received after an animation has become ready but before it started playing. In this case, the new priority etc. overwrites the previous values.

3 Configuring the SEMAINE system

Many aspects of the SEMAINE system can be configured. This section is intended as an initial pointer to the configurable parts of the system; more detailed information will be made available as part of the deliverable report D7b in December 2010.

3.1 System manager and java component runner

The starting point for configuration of the Java subsystem is a system config file, such as SEMAINE-3.0/java/config/speech2face.config. It lists the components to be loaded, can contain any system properties that may be accessed by the java components, and in particular points to additional config files.

Specifically:

- semaine.components lists the components to be loaded
- semaine.character-config points to the character config file (see below)
- semaine.stateinfo-config points to the state information config file (see below)
- semaine.DM-config points to the dialogue manager config file (see below).

3.2 Character config file

The character config file (e.g., SEMAINE-3.0/java/config/character-config.xml) contains the definition of the characters' properties, including the TTS voices they can use, their emotional predispositions, and their propensity to take the turn.

In a future version this file may also refer to the facial models that should be used for the visual appearance of the character, as well as any other character properties.

3.3 State info config file

The stateinfo config file (e.g., SEMAINE-3.0/java/config/stateinfo.config) is the backbone of communicating state information between the system components. It defines the short names of any information items – anything the system knows about the current state of the user, the agent, the dialog, and the context –, and defines how they are encoded and decoded in XML for communicating state within the system.

In order to use a new information item in the code, it is sufficient to add it to the stateinfo.config file and make sure all producers and consumers of this information use the revised stateinfo.config file.

C++ components that use state information currently expect stateinfo.config to be in the same folder as the executable binary.

3.4 Dialog manager config file

The dialog manager config file (e.g., SEMAINE-3.0/java/config/DM.config) identifies the dialogue templates that are used to drive the verbal behaviour of the agents. By adding or removing template files from the entry “template_files”, the user can change the dialog strategies used.

For example, depending on the intended steps when changing from one character to another, exactly one of the following three config files should be included in the templates list. After finishing a dialog with one of the SAL characters:

- /eu/semaine/components/dialogue/data/templates/CharChangeModeratorEval.xml will bring up a moderator character asking evaluation questions about the user's perception of the quality of the interaction, and then introduce the next character;
- /eu/semaine/components/dialogue/data/templates/CharChangeModerator.xml will bring up the moderator character who will directly introduce the next character;
- /eu/semaine/components/dialogue/data/templates/CharChange.xml will directly change from one character to the next without showing the moderator as intermediary.

The paths shown are interpreted as classpath locations, i.e. the respective files are expected to be in one of the jar files loaded when starting the system or as substructures in a directory that is included in the classpath when starting the system.

3.5 Listener behaviour

An xml file is used to configure the listener behaviour system. It is located in SEMAINE-3.0/Greta/listener/ASconfig.xml. It contains entries such as:

```
<character name="Poppy" mimicry="0.5" backchannel="0.5" noutterance="1">
    <respondTo head="true" face="true" acoustic="true"/>
</character>
```

For each character it defines the probability of generating mimicry, response backchannel (based on the agent's mental state) and utterances.

In order to switch off mimicry, set the mimicry attribute to 0 and the backchannel to 1. Conversely, to generate only mimicry behaviour, set the mimicry attribute to 1 and the backchannel to 0. Keep them at 0.5 for a similar quantity of mimicry and reponse backchannels.

The utterance attribute allows you to block the utterances coming from the dialogue manager. It is not a mandatory tag and if it is not there it is automatically set at 1 (all sentences are let through).

The agent's responsiveness to head, face or acoustic signals is determined by the attributes of the <respondsTo> element. Setting an attribute to "true" means that the listener intent planner generates signals for a certain modality (head, face, acoustic). This tag can be used without looking into the rules.

The individual rules that trigger and determine listener behaviour are quite straightforward. They describe the signals the agent reacts to and how. They are located in SEMAINE-3.0/Greta/listener/rulesfile.xml. For example:

```
<rule name="trigger-AU12">
    <usersignals>
        <usersignal id="1" name="AU12" modality="face"/>
    </usersignals>
    <backchannels probability="1.0" priority="2">
        <mimicry probability="0.6">
            <mimicry_signal name="mouth=smile" modality="face"/>
        </mimicry>
        <response_reactive probability="0.4"/>
    </backchannels>
</rule>
```

This rule is triggered when the AU12 is detected. It can generate a signal of mimicry (that will be a smile on the face modality) or a response backchannel. The probabilities in the rules should be ignored: they are not used anymore since the action selection has been implemented.

So, in order to avoid that an agent responds to a signal, it suffices to just delete the associated rule.

3.6 Speech input component configurations

The technical details of the opensmile system are determined by a config file such as SEMAINE-3.0/Opensmile/conf/opensmileSemaine3a.conf or c++/src/tum/auxiliary/conf/opensmileSemaine3a.conf (in the source release or the SVN trunk version). The actual config file used is included in the call to Opensmile's SEMAINExtract executable, e.g. in SEMAINE-3.0/Opensmile/semaine-openSMILE-win5-run.bat (for Windows XP systems) and SEMAINE-3.0/Opensmile/semaine-openSMILE-win6-run.bat (for Windows Vista and above) or in bin/semaine-openSMILE-run.bat (in the source release or the SVN trunk version).

The top-level configuration file includes several other configuration files, which cover individual sub-tasks. Comments in the file explain which includes need to be enabled in order to run which configuration. A few most important examples are illustrated here:

- **Voice activity detector**

  Two voice activity detectors exist. A simple detector which used a fixed signal energy threshold can be enabled via the configuration file opensmileSemaineVADsimple.conf. The threshold for the RMS frame energy must be adjusted in the file opensmileSemaineVADsimple.conf to your setup by changing the "threshold = xxxx" option in the section [turn:cTurnDetector]. Typical values range from 0.001 to 0.1.

  The advanced, self adapting voice activity detector is configured via opensmileSemaineVAD2.conf. Details are found in the D2b report. If the agent voice from the speakers causes problems (feedback), after the system is running for a certain time, uncomment the line "alwaysRejectAgent = 1" (remove the ";") in the section [vad:cRnnVad] in opensmileSemaineVAD2.conf.

- **Speech recognition**

  By default the single stream HMM only recogniser is enabled. On faster systems one may want to try the multi-stream LSTM/HMM hybrid architecture. This can be enabled by uncommenting the line

  ;\{opensmileSemaineASRms.conf}

  in opensmileSemaine3a.conf. Be sure to disable the single stream recogniser by commenting out the line which includes opensmileSemaineASR.conf.

  To disable the speech recognition (words and non-linguistics), comment out both opensmileSemaineASR.conf and opensmileSemaineASRms.conf.

- **Emotion recognition**

  The emotion recognition is split to three configuration files: feature extraction (e.g. conf_B/opensmileSemaineEmoftAc.conf for feature Set B – see D3c for details on the feature sets), dimensional emotion recognition (e.g. conf_B/opensmileSemaineEmoBling.conf

for acoustic and linguistic features or conf_B/opensmileSemaineEmoBsel.conf for acoustic features only), and detection of the user's level of interest (e.g. conf_B/opensmileSemaineIntB.conf). To disable either the dimensional affect recognition or the interest recognition, comment out the corresponding line. If both are disabled the line including the emotion feature extraction configuration should also be commented out.

Note: The feature extraction, dimensional affect recognition and interest detection configuration files from different feature sets (A, B, and C) may not be mixed.

Futher, the SEMAINExtract executable supports the command-line options -noWords, -noNonverbals, and -noInterest, to selectively disable the semaine components and their functionality as they appear in the GUI (Note, that the information is still computed and displayed in the openSMILE debug output, however it is not sent to the semaine system if the component is disabled – on the other hand, if the extraction of certain things is disabled in the opensmileSemaine3a.conf file, the the information will not be sent, even if the sender component is enabled and thus still visible).

3.7 Video input component configurations

The video input components, when installed, can be configured using C:\Program Files\iBUG\Semaine Video Components\videoConfig.cfg. Among other things, this config file determines whether a USB or Firewire camera is used.

3.8 MARY TTS configuration

The speech synthesis components are configured using a number of configuration files in SEMAINE-3.0/MARY/conf. The most important config file is marybase.config; its most important setting is “cache = false”. In order to switch on TTS caching to speed up the system, change this to “cache = true”.

Another essential means of configuring the MARY TTS system is by adding or removing voices. As a rule of thumb, the more voices are installed, the slower the system becomes, so handle with care. Voices are installed and uninstalled using the MARY component installer which is located in SEMAINE-3.0/MARY/bin.

4 License and availability

The SEMAINE API for Java and C++, the SEMAINE dialogue components (in Java), and the speech synthesizer MARY TTS are distributed under the GNU Lesser General Public License (LGPL), version 3. The speech synthesis voices for the SAL agents are distributed under the Creative Commons ShareAlike - No Derivatives license.

The 3D agent animation software Greta and the speech analysis software Opensmile are distributed under the GNU General Public License (GPL).

The separately installable SEMAINE Video components for camera image analysis come as a freeware binary.

References

Johnston, M., Baggia, P., Burnett, D. C., Carter, J., Dahl, D. A., McCobb, G., & Raggett, D. (2009, February 10). EMMA: Extensible MultiModal Annotation markup language. W3C Recommendation. Retrieved from http://www.w3.org/TR/emma/

Schröder, M. (2010). The SEMAINE API: Towards a standards-based framework for building emotion-oriented systems. *Advances in Human-Computer Interaction*, *2010*(319406). doi:10.1155/2010/319406

Schröder, M., Baggia, P., Burkhardt, F., Oltramari, A., Pelachaud, C., Peter, C., & Zovato, E. (2010). *Emotion Markup Language (EmotionML) 1.0* (W3C Working Draft). World Wide Web Consortium. Retrieved from http://www.w3.org/TR/2010/WD-emotionml-20100729/

Appendix A: stateinfo.config

The stateinfo.config file defines short names and how to encode the information for passing state messages.

```
# This is the config file for user state info messages in the SEMAINE project.
# It provides for each piece of information that can be represented a simple XPath
expression
# describing how to encode this information in state info messages.

######################################################################

[UserState]

[namespace prefixes]
semaine http://www.semaine-project.eu/semaineml
bml     http://www.mindmakers.org/projects/BML
emotion http://www.w3.org/2009/10/emotionml


[short names]
headGesture   /semaine:user-state/bml:bml/bml:head/@type
headGestureStarted  /semaine:user-state/bml:bml/bml:head/@start
headGestureStopped  /semaine:user-state/bml:bml/bml:head/@end
facialExpression    /semaine:user-state/bml:bml/bml:face/@shape
facialActionUnits   /semaine:user-state/bml:bml/bml:face/@au
facialExpressionStarted    /semaine:user-state/bml:bml/bml:face[@shape]/@start
facialExpressionStopped    /semaine:user-state/bml:bml/bml:face[@shape]/@end
facialActionUnitsStarted   /semaine:user-state/bml:bml/bml:face[@au]/@start
facialActionUnitsStopped   /semaine:user-state/bml:bml/bml:face[@au]/@end
pitchDirection      /semaine:user-state/semaine:pitch/@direction
speaking     /semaine:user-state/semaine:speaking/@status

vocalization /semaine:user-state/semaine:vocalization/@name
facePresent /semaine:user-state/semaine:face-present/@status

interest     /semaine:user-state/emotion:emotion[@dimension-set='http://www.semaine-
project.eu/emo/dimension/interest.xml']/emotion:dimension[@name='interest']/@value
valence /semaine:user-state/emotion:emotion[@dimension-
set='http://www.example.com/emotion/dimension/FSRE.xml']/emotion:dimension[@name='valence
']/@value
arousal /semaine:user-state/emotion:emotion[@dimension-
set='http://www.example.com/emotion/dimension/FSRE.xml']/emotion:dimension[@name='arousal
']/@value
potency /semaine:user-state/emotion:emotion[@dimension-
set='http://www.example.com/emotion/dimension/FSRE.xml']/emotion:dimension[@name='potency
']/@value
unpredictability /semaine:user-state/emotion:emotion[@dimension-
set='http://www.example.com/emotion/dimension/FSRE.xml']/emotion:dimension[@name='unpredi
ctability']/@value
intensity /semaine:user-state/emotion:emotion/emotion:intensity/@value
emotion-quadrant /semaine:user-state/emotion:emotion[@category-set='http://www.semaine-
project.eu/emo/category/four-quadrants.xml']/emotion:category/@name
userUtterance       /semaine:user-state/semaine:userutterance/@utterance
userUtteranceStartTime     /semaine:user-state/semaine:userutterance/@starttime
userUtteranceFeatures      /semaine:user-state/semaine:userutterance/@features
gender       /semaine:user-state/semaine:gender/@name

######################################################################

[AgentState]

[namespace prefixes]
```

```
semaine http://www.semaine-project.eu/semaineml
emotion http://www.w3.org/2009/10/emotionml

[short names]
needToSpeak    /semaine:agent-state/semaine:needtospeak/@value
turnTakingIntention        /semaine:agent-state/semaine:turntakingintention/@value

# The following are bipolar scales, value range from 0 to 1; values below 0.5 mean
disagreement etc., values above 0.5 mean agreement etc.
# They stem from research by Elisabetta Bevacqua et al.
# disagreement-agreement
agreement /semaine:agent-state/emotion:emotion[@dimension-set='http://www.semaine-
project.eu/emo/dimension/communicative-
functions.xml']/emotion:dimension[@name='agreement']/@value
# refusal-acceptance
acceptance /semaine:agent-state/emotion:emotion[@dimension-set='http://www.semaine-
project.eu/emo/dimension/communicative-
functions.xml']/emotion:dimension[@name='acceptance']/@value
# disbelief-belief
belief /semaine:agent-state/emotion:emotion[@dimension-set='http://www.semaine-
project.eu/emo/dimension/communicative-
functions.xml']/emotion:dimension[@name='belief']/@value
# disliking-liking
liking /semaine:agent-state/emotion:emotion[@dimension-set='http://www.semaine-
project.eu/emo/dimension/communicative-
functions.xml']/emotion:dimension[@name='liking']/@value
# no-understanding-understanding
understanding /semaine:agent-state/emotion:emotion[@dimension-set='http://www.semaine-
project.eu/emo/dimension/communicative-
functions.xml']/emotion:dimension[@name='understanding']/@value
# no-interest-interest
interest /semaine:agent-state/emotion:emotion[@dimension-set='http://www.semaine-
project.eu/emo/dimension/communicative-
functions.xml']/emotion:dimension[@name='interest']/@value

# The following are emotional, cognitive and interactional dimensions found to be
relevant for
# listener vocalisations by Sathish Pammi et al.
# - unipolar dimensions, from "not present" to "intensely present":
anger /semaine:agent-state/emotion:emotion[@dimension-set='http://www.semaine-
project.eu/emo/dimension/listener-meanings.xml']/emotion:dimension[@name='anger']/@value
sadness /semaine:agent-state/emotion:emotion[@dimension-set='http://www.semaine-
project.eu/emo/dimension/listener-
meanings.xml']/emotion:dimension[@name='sadness']/@value
amusement /semaine:agent-state/emotion:emotion[@dimension-set='http://www.semaine-
project.eu/emo/dimension/listener-
meanings.xml']/emotion:dimension[@name='amusement']/@value
happiness /semaine:agent-state/emotion:emotion[@dimension-set='http://www.semaine-
project.eu/emo/dimension/listener-
meanings.xml']/emotion:dimension[@name='happiness']/@value
contempt /semaine:agent-state/emotion:emotion[@dimension-set='http://www.semaine-
project.eu/emo/dimension/listener-
meanings.xml']/emotion:dimension[@name='contempt']/@value
# - bipolar dimensions in the sense that both end points are marked:
# low-to-high anticipation
anticipation /semaine:agent-state/emotion:emotion[@dimension-set='http://www.semaine-
project.eu/emo/dimension/listener-
meanings.xml']/emotion:dimension[@name='anticipation']/@value
# low-to-high solidarity
solidarity /semaine:agent-state/emotion:emotion[@dimension-set='http://www.semaine-
project.eu/emo/dimension/listener-
meanings.xml']/emotion:dimension[@name='solidarity']/@value
# low-to-high antagonism
antagonism /semaine:agent-state/emotion:emotion[@dimension-set='http://www.semaine-
project.eu/emo/dimension/listener-
```

```
meanings.xml']/emotion:dimension[@name='antagonism']/@value
# The remaining dimensions from Pammi's work are considered to be the same as some of the
communicative functions above, namely:
# uncertain-certain => disbelief-belief
# disagreeing-agreeing => disagreement-agreement
# uninterested-interested => no-interest-interest


######################################################################

[DialogState]
[namespace prefixes]
semaine http://www.semaine-project.eu/semaineml

[short names]
userTurnState        /semaine:dialog-state/semaine:user/@believesHasTurn
agentTurnState      /semaine:dialog-state/semaine:agent/@believesHasTurn
convState    /semaine:dialog-state/semaine:agent/@convState

######################################################################

[ContextState]
[namespace prefixes]
semaine http://www.semaine-project.eu/semaineml

[short names]
userPresent   /semaine:situational-context/semaine:user/@status
character    /semaine:situational-context/semaine:character/@name
nextCharacter       /semaine:situational-context/semaine:character/@next
dialogContext       /semaine:situational-context/semaine:dialog-context/@name

######################################################################

[SystemState]
[namespace prefixes]
semaine http://www.semaine-project.eu/semaineml

[short names]
cameraPresent        /semaine:setup/semaine:camera/@status
cameraXResolution   /semaine:setup/semaine:camera/@xres
cameraYResolution   /semaine:setup/semaine:camera/@yres
cameraFrameRate     /semaine:setup/semaine:camera/@framerate
cameraNumChannels   /semaine:setup/semaine:camera/@numChannels
microphonePresent   /semaine:setup/semaine:microphone/@status
microphoneFrameRate       /semaine:setup/semaine:microphone/@framerate
ecaPresent   /semaine:setup/semaine:eca/@status
```