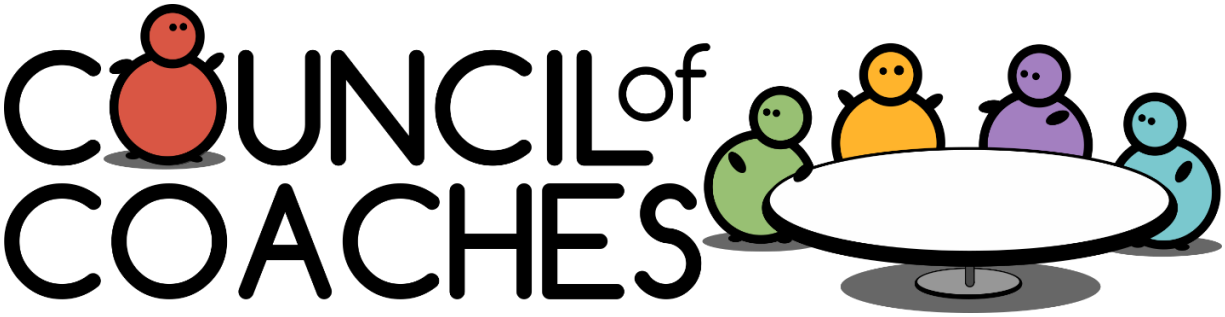

D7.3: Second functional prototype

Dissemination level: Public

Document type: Demonstrator

Version: 1.0.1

Date: January15, 2019 (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	Second functional prototype
Due date of deliverable	November 30, 2018
Work package	WP7
Author(s)	Dennis Reidsma (CMC), Merijn Bruijnes (CMC), Gerwin Huizing (CMC), Randy Klaassen (CMC), Kostas Konsolakis (CMC), Marcel Weusthof (CMC), Jorien van Loon (CMC), Tessa Beinema (RRD), Silke ter Stal (RRD), Dennis Hofs (RRD), Boris van Schooten (RRD), Harm op den Akker (RRD), Mark Snaith (UDun), María Jesus Arnal (UPV), Jose Luís Bayo (UPV), Álvaro Fides (UPV), Reshmashree Bangalore Kantharaju (SU), Donatella Simonetti (SU)
Reviewer(s)	Harm op den Akker (RRD)
Approved by	Coordinator
Dissemination level	PU: Public
Document type	Demonstrator
Total number of pages	21

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, Grupo SABIEN (UPV)
- Innovation Sprint (iSPRINT)

Abstract

A description of the second version of the Functional Prototype in its two variants: the design-oriented Functional Prototype used to gather feedback from users and the Technical Prototype where all the technical integration progress is consolidated. This second version already contains updates based on the feedback gathered by WP2 and evaluations with users. This document also reports the established development and issue management processes.

Corrections

v1.0.1 Correctly applied EU logo on header page.

Table of Contents

List of figures

List of tables

Symbols, abbreviations and acronyms

ASAP	Articulated Social Agents Platform
BML	Behaviour Markup Language
CMC	Centre for Monitoring and Coaching
COUCH	Council of Coaches
D	Deliverable
DBT	Danish Board of Technology Foundation
DGEP	Dialogue Game Execution Platform
EC	European Commission
ISPRINT	Innovation Sprint
M	Month
MS	Milestone
RRD	Roessingh Research and Development
RRI	Responsible Research and Innovation
SU	Sorbonne University
UDun	University of Dundee
UPV	Universitat Politècnica de València
UT	University of Twente
WP	Work Package

1 Introduction

This document of type "Demonstrator" is accompanying the second prototype release of the Council of Coaches project. In Council of Coaches, we distinguish between the "Functional Demonstrator" track, and the "Technical Demonstrator" track. The Functional Demonstrator provides a working, stable, "production ready", robust demonstrator that can be evaluated with older adult end-users, while the Technical Demonstrator contains more state-of-the-art features, that is at the "lab" level of readiness.

This document accompanies the release of the two software demonstrator in the following way:

- Section 3 provides details and background information on the **development process**, and continues to explain the rationale behind the functional/technical demonstrator division.
- Section 4 provides a quick walkthrough of the **Functional Demonstrator** – and links to the working prototype that is available online.
- Section 5 provides the technical instructions necessary to run the **Technical Demonstrator**.

2 Objectives

This is the second in a collection of deliverables reporting the progress of the prototype system of Council of Coaches:

- **D7.2: Initial functional prototype (M9):** An early low-fidelity prototype of the system, using Wizard-of-Oz methodologies to compensate for functionalities still under development.
- **D7.3: Second functional prototype (M15):**(This document) Focused on delivering a more realistic and smooth user experience, including a major update of the contents (dialogue possibilities).
- **D7.4: Third functional prototype (M21):** Used to test acceptance and usability of more advanced features (agent animations and behaviours, interaction concepts).
- **D7.5: Final Council of Coaches Technical Prototype (M27):** Include all the innovations connecting Shared Knowledge Base, Holistic Sensing Framework with autonomous Dialogue Framework, and Embodied Conversational Agents to deliver a fully working system.

From the reporting perspective, this document acts as a reference pointing to all the actual source code, binaries, executables, the software, and accompanying instructions that build up the Second Functional Prototype. Altogether, this deliverable document and the referenced material, are the delivered Second Functional Prototype.

In order to achieve this, this document targets two objectives:

Objective 1: To report the work done in creating the second prototype since the release of the first functional prototype until the release date of the second.

Objective 2: To provide links to the Second Functional Prototype software and document how to execute it.

3 Development process

3.1 Development management

In the previous version of this deliverable (D7.2) we reported that we would use GitLab as the code-hosting solution for Council of Coaches. The Council of Coaches project was already set up in GitLab back then (https://gitlab.com/CouncilOfCoaches). It is currently in a private status, so that only members of the project can see it. Since that time we have continued to take advantage of the tools provided by GitLab, and have integrated them into an AGILE-inspired development management process.

A periodic teleconference has been scheduled every two weeks, with the presence of at least one representative of each WP and partner, where they report:

- What work has been done since last call.
- What problems they have encountered.
- What will be worked on until next call.
- Questions and any other business.

To keep a formal track of the work done, a list of Issues has been set up in GitLab (https://gitlab.com/groups/CouncilOfCoaches/-/issues) to be tracked through an Issue Tracker Board (https://gitlab.com/groups/CouncilOfCoaches/-/boards) with 4 lists:

- *Open*: Backlog of all the currently open and unaddressed issues.
- *To Do*: Issues to address before the next Milestone (Council of coaches > Issues > Milestones). Move them here from Backlog when their Milestone period begins.
- *Doing*: Move issues here when someone starts actively working them.
- *Closed*: Implemented, solved or deprecated issues. It is possible to reopen them.

Project Milestones (https://gitlab.com/groups/CouncilOfCoaches/-/milestones) have been created that match the different versions of the Prototype. Issues can be classified along with their management procedures in Requirements, Bugs and Others.

Requirements

These are imported from the architecture Requirements Viewpoint and updated as new requirements are identified from user feedback. WP7 management will add them whenever there is an official update to the requirements. Labels include either *Functional* or *Non-Functional*, and the related WP that has to implement it. Assignees will be set when appropriate.

When it is clear when the requirements will be implemented, it will be assigned a Milestone. When the Milestone is the current one, the requirement issue will be added to the current *To Do* list. When work begins on a requirement it will be added to the *Doing* list and closed when it is implemented.

Bugs

These issues report bugs encountered when using the code. Any member of the project can create a Bug issue when it is found. The Title must be concise and descriptive. The Labels must include *Bug*. The Description must include as much information as possible: What the bug is, how to reproduce it, what should have happened instead, and any logs, errors or attachments that are helpful. If a responsible can be identified, they will be added as the Assignee. Otherwise WP7 management will assign the appropriate developer.

When a developer is assigned a Bug issue, they must try to reproduce it, and use the comments to request feedback from the reporter. If it is not reproducible, does not qualify as bug, or is fixed immediately, the issue will be closed. Otherwise, it will be classified as:

- *Minor*: Has to be fixed at some point before the end of the project.
- *Major*: Has to be fixed before the next milestone. Add that Milestone reference to this issue. Add this issue to the current *To Do* list.
- *Critical*: Has to be fixed ASAP. Add current Milestone reference to this issue. Add this issue to the current *To Do* list, and then *Doing* list.

Others

Anyone can create (or be assigned, or get involved in) any other type of issue. These issues must provide a clear Title and Description and the appropriate generic Label:

- *Support*: Ask for help for running or using the code. This may eventually lead to a bug.
- *Enhancement*: How to improve an already developed feature or requirement.
- *Suggestion*: A new requirement or fix for a bug that we have not thought about yet.
- *Documentation*: Request to create or improve code documentation.
- *Discussion*: Feel free to comment any related topic.

3.2 Functional Prototype and Technical Prototype

As reported in the previous version (D7.2), the development effort of the prototype was split in two in order to accommodate the early needs for evaluation. This resulted in two interlinked "builds":

Technical Prototype: This is the core of the Council of Coaches code, which will eventually become the final solution. Development is ongoing, as pre-existing technical solutions are being integrated by the Technical Integration Taskforce, and new modules are being designed, developed and added to satisfy all pending requirements. As a result, the output of this build is far from refined and may be counter-productive to use it for evaluation of elements (such as user interface or experience) that have not yet been finalized. This is what the Functional Prototype is used for.

Functional Prototype: This is a reduced, or "lite" version of Council of Coaches that is built on a basic build of the Technical Prototype. Unfinished features are replaced with temporary modules, such as a simplified web-based user interface. This allows the Functional Prototype to be used early on in evaluation sessions to gather feedback from users, and then feed it into the development of the Technical Prototype. As the Technical Prototype gets enhanced with this feedback and the remaining features, subsequent Functional Prototypes will incorporate more parts of it, until being finally superseded by it in the final iteration.

3.3 Technical Integration Taskforce

The Technical Integration Taskforce (introduced in D7.2) has proven valuable as it managed to integrate existing software into the working First Functional Prototype. Therefore the taskforce will continue as the main driving force behind the Technical Prototype. After all it is comprised of the bulk of the developer staff of the project. As the release date of each Functional Prototype approaches it will be spun up from the Technical Prototype where the taskforce build the major project developments. Since the last version of this deliverable, work has continued in this regard, further integrating the separate modules and designing and creating new ones identified in the architecture. The architecture itself has been updated and the latest diagrams and models are available in our GitLab repository (https://gitlab.com/CouncilOfCoaches/Architecture). As part of the ongoing development work, technical Workshop Meetings have been useful, and a new meeting was held during this period.

3.3.1 Workshop 3

The third workshop meeting of the Technical Integration Taskforce was held in Enschede during the week of 8th-12th of October. The aim of this workshop was:

- Analyse the results of the evaluation of the First Functional Prototype and identify requirements
- Determine enhanced scenarios for the Second Functional Prototype
- Layout the development tasks until the release of the Second Functional Prototype
- An updated RRI mini-workshop to track new and pending RRI issues

The concrete technical outcomes of this workshop were:

- Definition of concrete age-related scenario for the Second Functional Prototype
- Inclusion of Greta agent into Unity agents along with embedded browser
- Enhanced sensor data gathering and native integration with FIWARE platform
- Design of data models and interfaces for the Knowledge Base
- Conversion of Yarn to DGDL
- Enhanced gaze behaviour

4 Functional Prototype software

4.1 Technology

The Council of Coaches functional demonstrator is a technologically simplified version of Council of Coaches that has as its main purpose to demonstrate the core concepts of providing coaching with an ensemble of virtual coaches on multiple domains. The functional demonstrator is a client-server tool, where the client is a website that can be accessed from any device. The location of the functional demonstrator website is given in Table 1 below.

Table 1: Web address for the Council of Coaches functional demonstrator.

http://demo.council-of-coaches.eu/

The web client is an HMTL/JavaScript client that communicates to a back-end server for each interaction between coach and user. In Figure 1 below, a high-level overview of the back-end server is given. An external REST API provides the window to the web client, managed by the *AgentController*. An *AgentManager* handles for each specific visitor to the web client the ongoing dialogues. Dialogues are written in the YARRDN language, a modification of the YARN dialogue language that includes additional control features for dynamic dialogues. The YARRDN Core is a library that can read and execute such dialogue definitions. The ability to store and use variables is integrated in the YARRDN dialogues and the Knowledge Base component is used to store values, and perform more complex operations, like calculating the outcomes of a finalized questionnaire dialogue. For persistence, the Knowledge Base is stored server side using a running CouchDB instance.

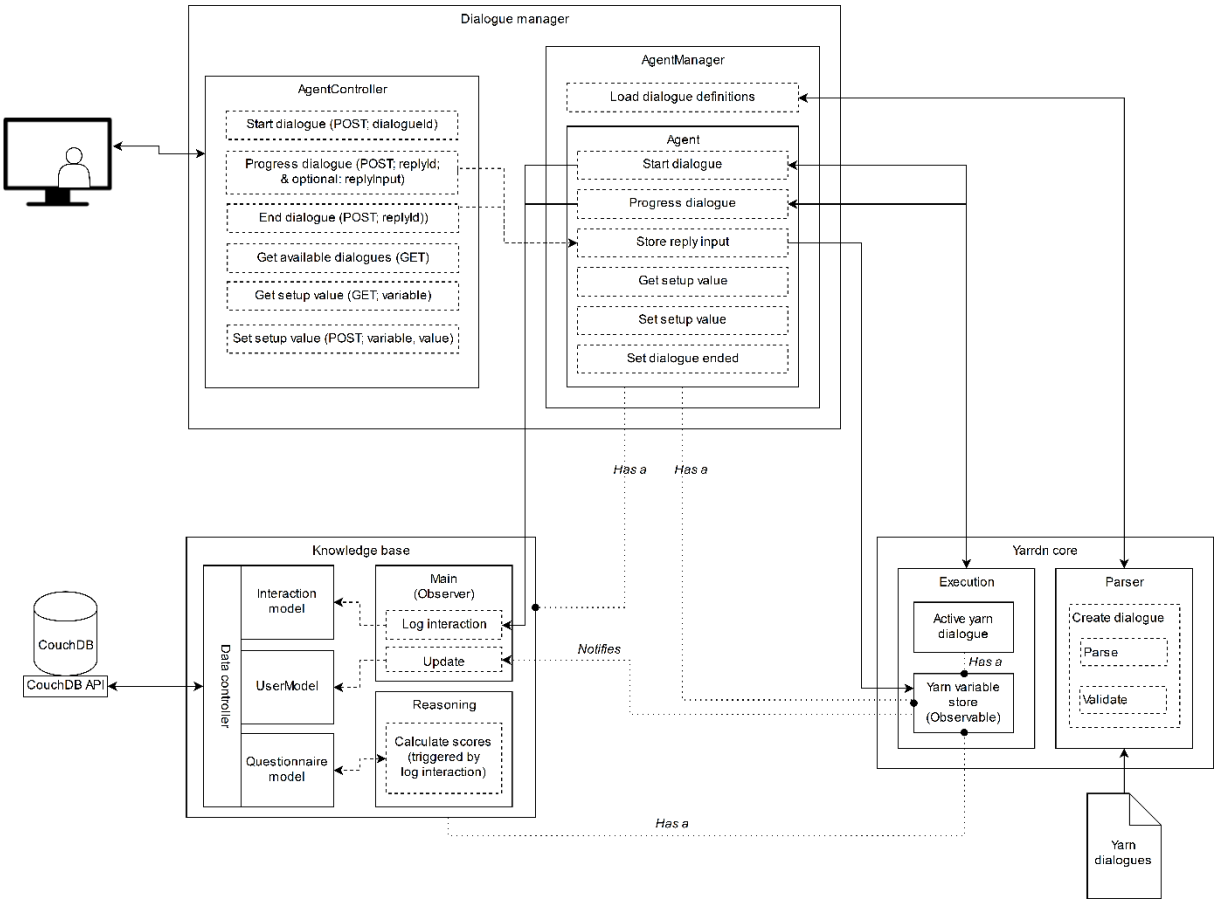

Figure 1: High level technology overview of the back-end server of the Council of Coaches functional demonstrator.

4.2 Function

Below, we describe a quick walkthrough through the contents of the functional demonstrator at the time of the M15 release. At the time of writing, the contents of this demonstrator were only available in Dutch – this walkthrough below provides an English language description of what is happening – pending the English translations of the demonstrator (Dutch is the primary language for the evaluation of this demonstrator).

At a first visit to the demonstrator – users are presented with the following screen (Figure 2), which acts as a "menu" that will guide the user through the various steps in the demonstrator. The first step, is to "Start the introduction", which can be accessed by clicking the "Start" button.

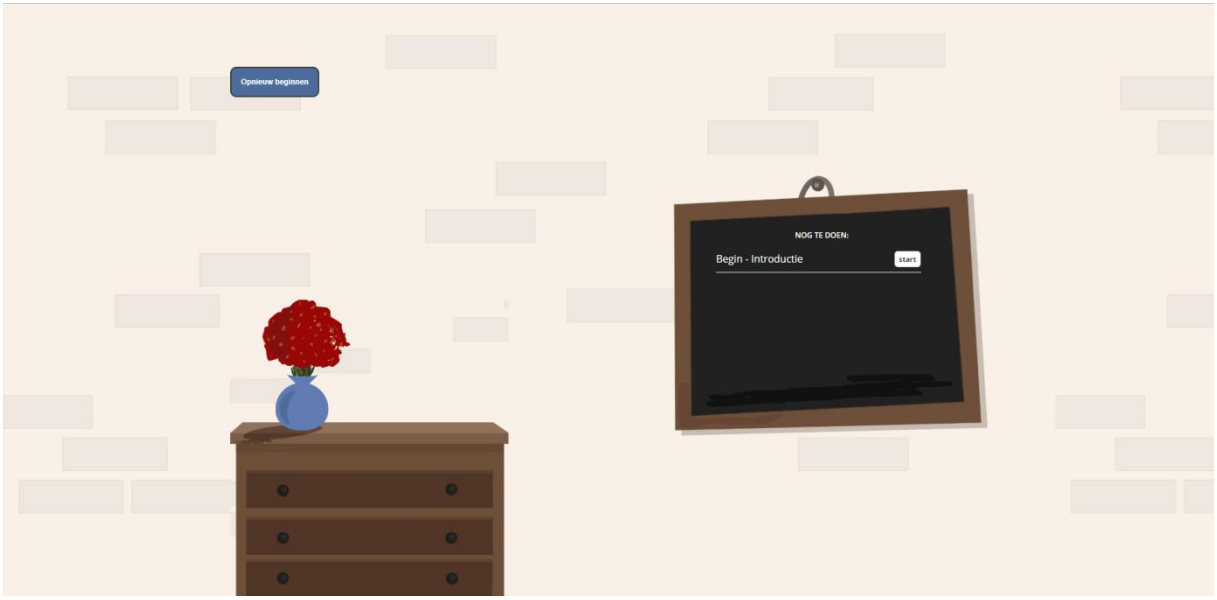

Figure 2: Start screen of the Second Functional Demonstrator.

Upon selecting the start button, the user starts his interaction with "Peter" – acting as the "head" of the Council of Coaches – someone that can explain the concept to the user, perform an initial intake and help in choosing the right coaches for the user.

In the current version of the functional demonstrator, Peter offers a short introductory dialogue, explaining what the Council of Coaches is, and explaining the user that as a first step, the intake questionnaire will be taken (Figure 3).

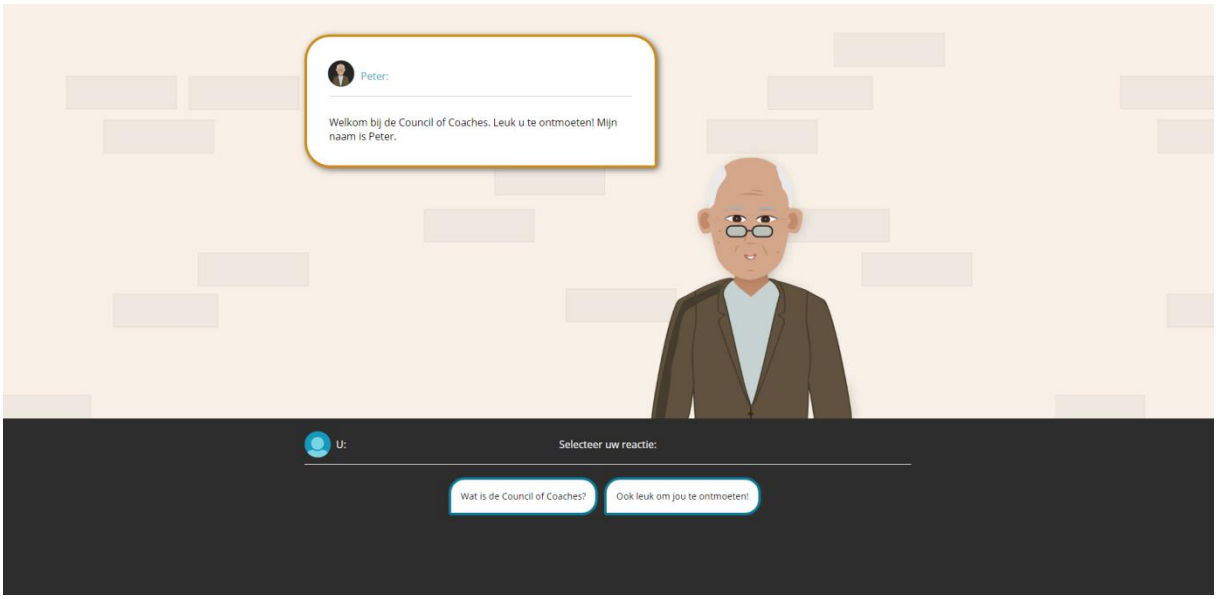

Figure 3: Peter offers a first introduction to the Council of Coaches and explains how a first intake will be done.

When the user indicates his readiness, the intake dialogue will start. In the current demonstrator, we have translated the SF-36 questionnaire into an interactive dialogue. SF-36 is a 36-item questionnaire that measures quality of life on various domains, and provides a good starting point for the selection of relevant coaches for the end user (see Figure 4).

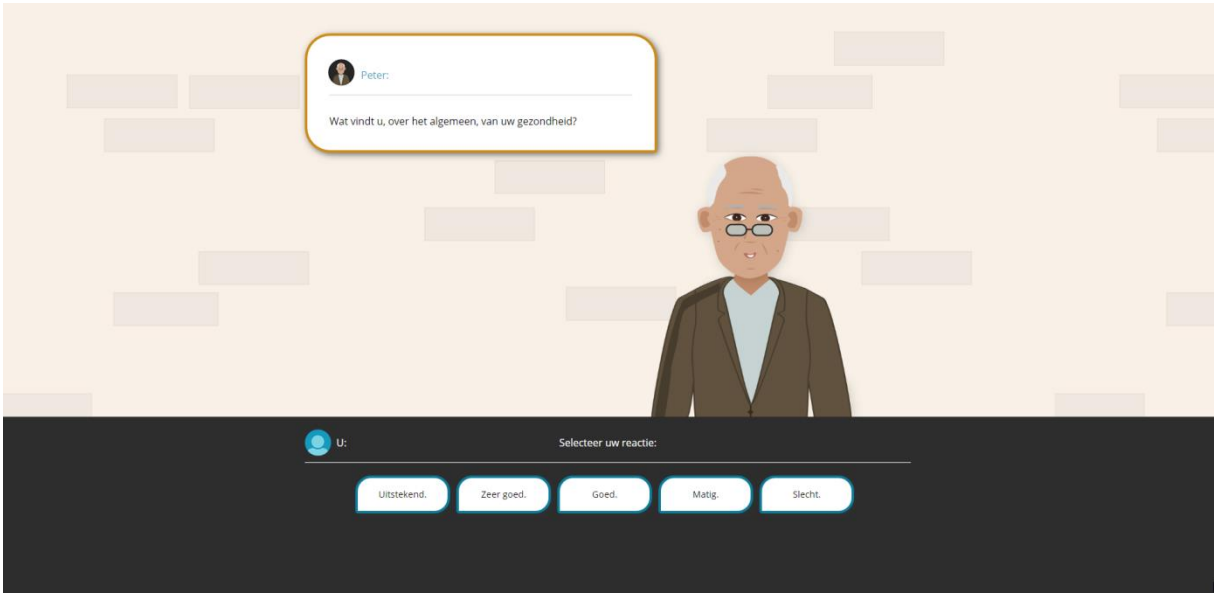

Figure 4: Peter takes the SF-36 questionnaire through an interactive dialogue.

After the user completes the SF-36 dialogue, he/she is taken back to the menu screen to select the next step in the demonstrator – getting to know the coaches (see Figure 5).

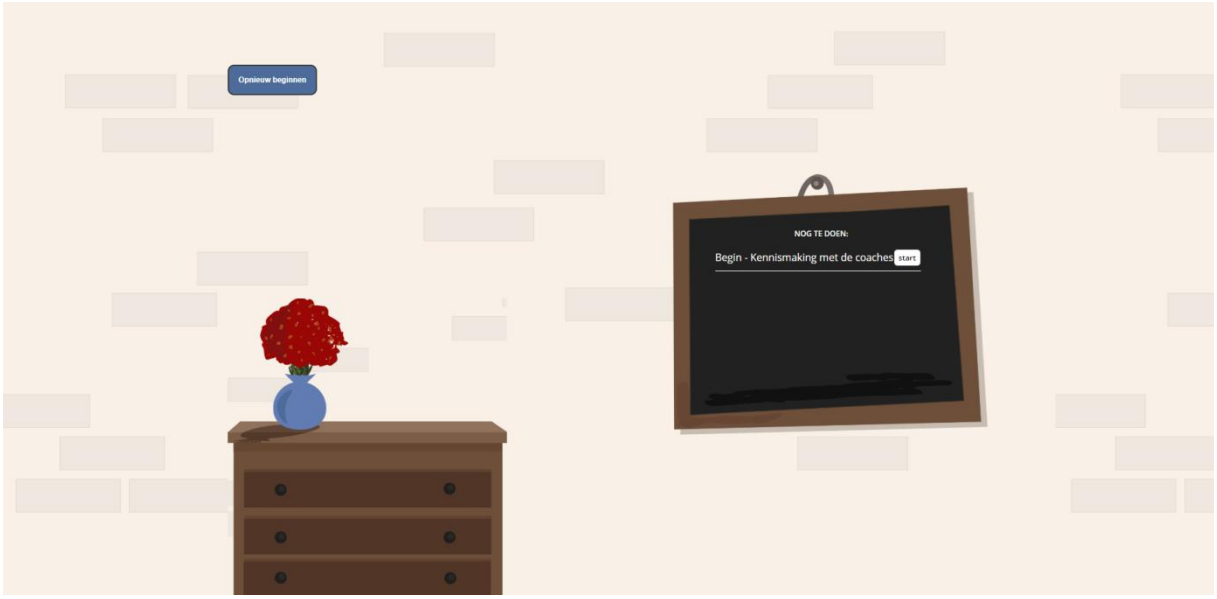

Figure 5: Back to the menu screen to start the next step "getting to know the coaches".

After pressing the "Start" button for the next item "getting to know the coaches", the user is introduced to the main Council of Coaches interface, where Peter is joined by three different coaches (see Figure 6).

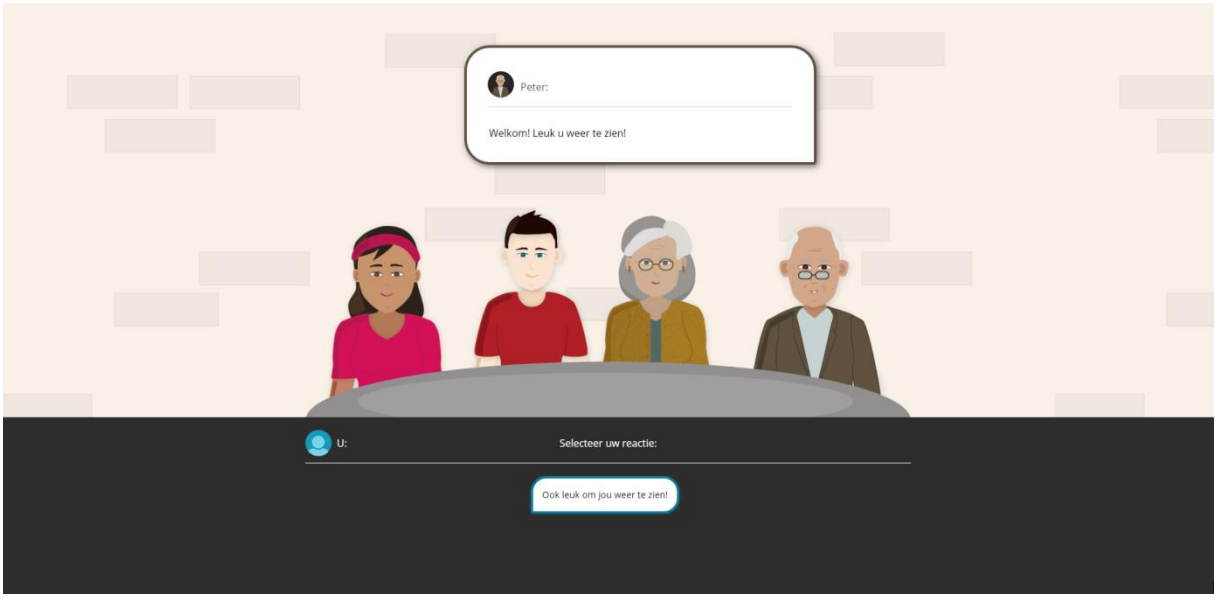

Figure 6: The main Council of Coaches interface where Peter is joined by Alexa (Physical Activity Coach), Francois (Diet Coach) and Helen (Cognitive Coach).

In this next part of the demonstrator, the user gets to talk to the three different coaches available in this version of the Council of Coaches: Alexa, the Physical Activity Coach, Francois, the Diet Coach, and Helen, the Cognitive Coach. The user can have short conversations with each of the three coaches.

Then, Peter explains that the coaches need to observe the user for a few weeks, before starting the actual coaching part. The coaches say goodbye, and hope to the see the user in a few weeks! This takes the user back to the menu screen to start the next part of the demonstrator (see Figure 7).

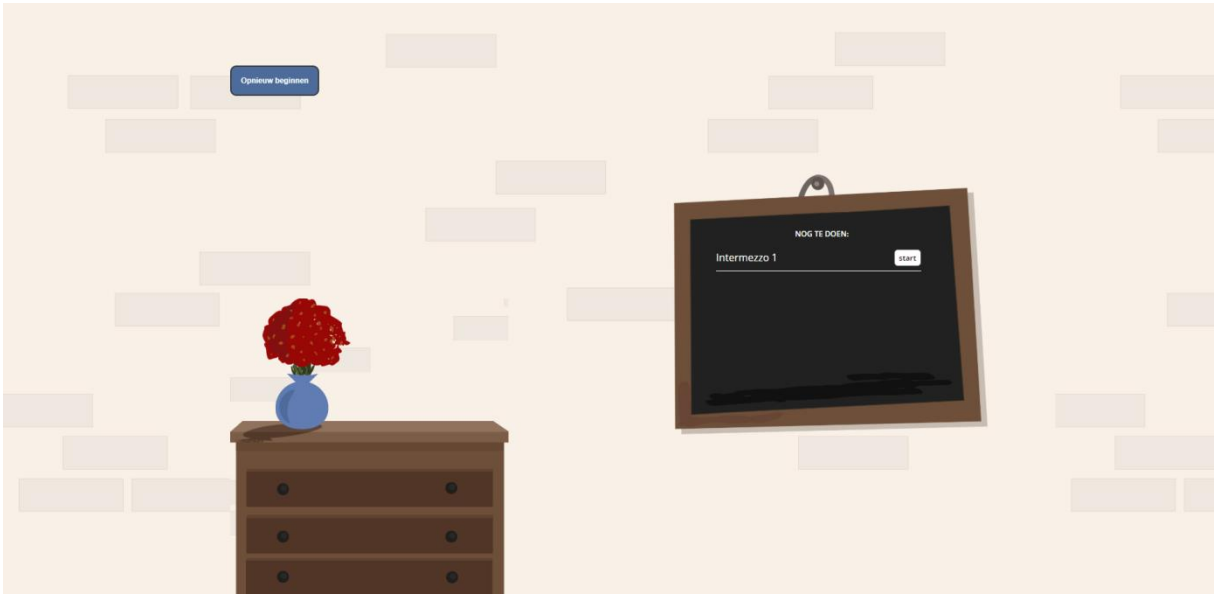

Figure 7: Menu screen to start the next phase, the short "Intermezzo".

The short “intermezzo” that the user can start consist of Peter, the main coach that explains the user that this demonstrator does not contain any actual measurements and instead asks the user to answer a few questions about their current behavior in the domain of daily physical activity, cognitive training, and diet. After completing this short intermezzo, the user is taken back to the main Council of Coaches screen (see Figure 8), in which Alexa, Francois and Helen now give coaching advice based on the provided answers during the intermezzo described earlier.

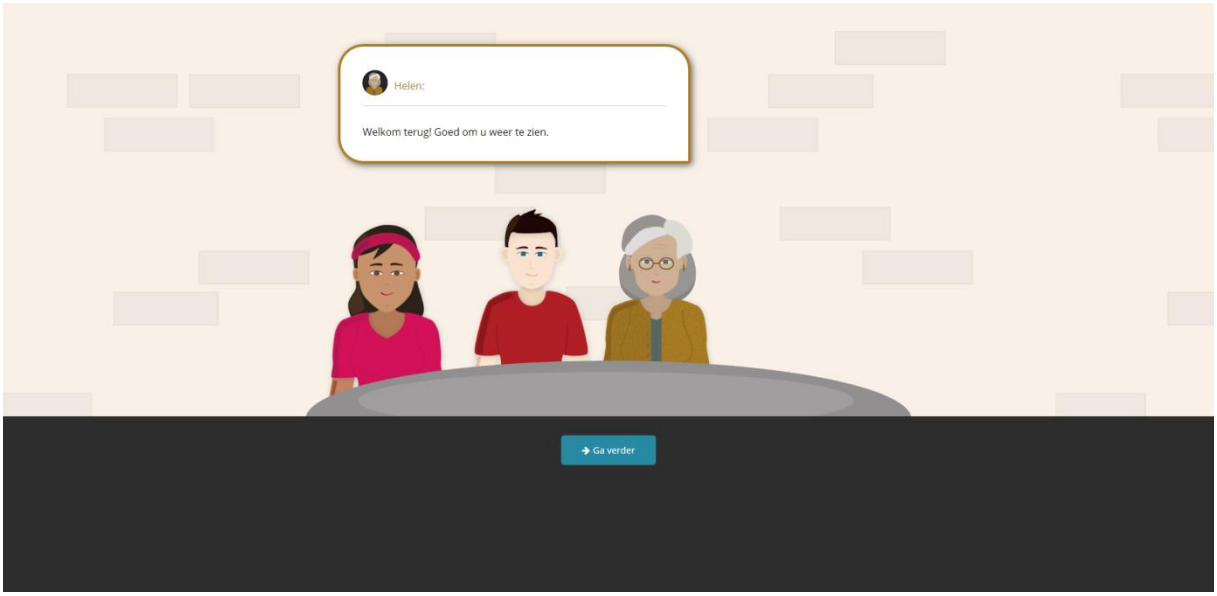

Figure 8: Back to the main Council of Coaches interface for some coaching advice.

5 Technical Prototype software

Unlike the previous version of this deliverable, it was not possible to prepare a one-click installation program in time. The standard developer installation instructions still apply: The following instructions are available in our project GitLab page: https://gitlab.com/CouncilOfCoaches/TechnicalDemonstrator . This is a summarized version.

To access our project in GitLab you will have to provide us with your GitLab account in order to grant permissions to access the project, since it is currently private.

Take into account that the first time the prerequisites and installation steps are followed it will take a considerable amount of time (over 1 hour).

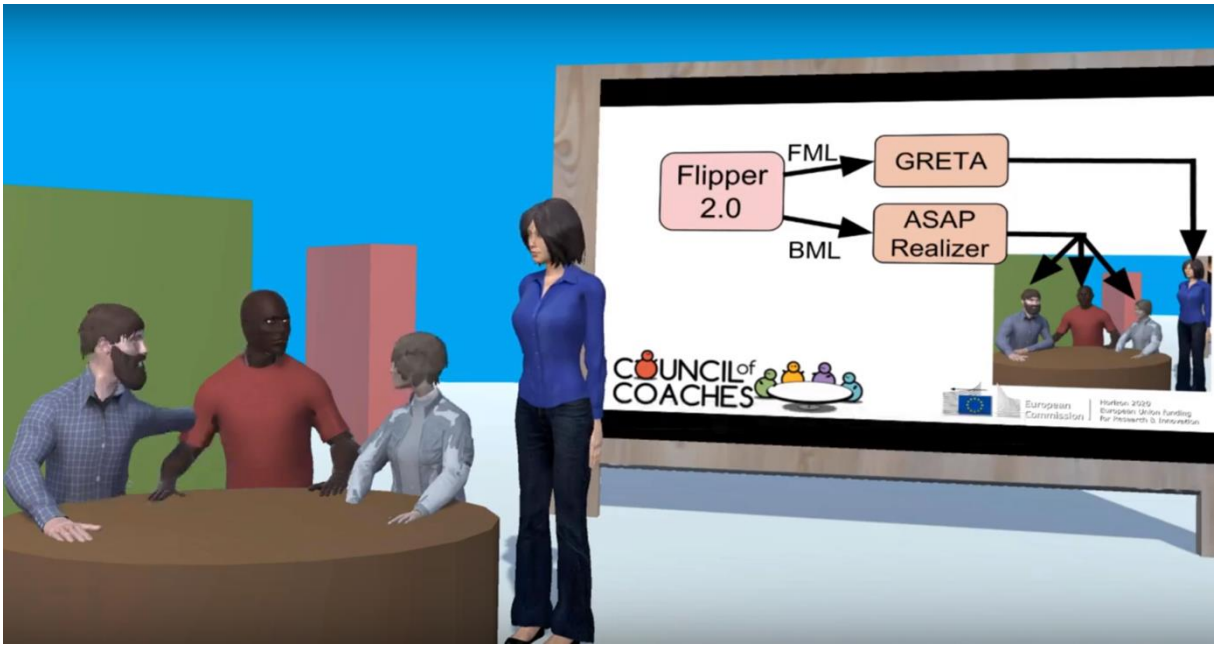

Figure 9: Technical Prototype Unity scene – three ASAP agents (leftmost agents), one GRETA agent (right agent) and the virtual web browser.

5.1 Pre-requisites

- Windows 10 (Pro) (Current instructions are only for Windows)
- Unity3D (version 2017.3.1f1 - other 2017.x versions might work, versions 2018.x currently do NOT work) https://unity3d.com/get-unity/download/archive
- Docker: https://www.docker.com/
- Java 8 JDK: http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html
- Visual C++ Redistributable for Visual Studio versions 2013 and/or 2015 (list available here: https://support.microsoft.com/en-gb/help/2977003/the-latest-supported-visual-c-downloads)
- Ant: https://ant.apache.org/
- Approximately 10GB of disk space

5.2 Installation

1. Clone (git@gitlab.com:CouncilOfCoaches/TechnicalDemonstrator.git) or download directly (https://gitlab.com/CouncilOfCoaches/TechnicalDemonstrator/-/archive/master/TechnicalDemonstrator-master.zip) the code fromour repository.

2. Start Docker. Configure through right-clicking tray icon >settings. Share the main disk drive. You may also want to increase allocated memory to 4GB.
3. Open a command line shell, go to *{installation}\UDUN_ArgTech* and type command *docker-compose pull*
4. Open a command line shell, go to *{installation}\UT_HMI\HmiCouch* and type command *ant resolve*
5. Open Unity, select *Open project*, and then select folder *{installation}\UT_HMI\UnityProject\CharacterCreatorNew*
6. In *project assets* (usually bottom-left), navigate to *\Assets\HMI\Scenes* and double-click the scene *CouchProjectDemo.unity*
7. In *project hierarchy* (usually left), search *GlobalAMQSettings* and in the details pane (usually right) set port to 61616.
8. It is possible that, depending on the version of Unity you have, there are some build errors. To fix them, go to *{installation}\UT_HMI\UnityProject\CharacterCreatorNew\Assets\UMA\Core\Extensions\DynamicCharacterSystem\UMAAssetBundleManager*and edit the files *BuildScript.cs* and *Utility.cs* to remove the *case*StandaloneOSX
9. Clone (https://github.com/gretaproject/greta.git) or download directly (https://github.com/gretaproject/greta/archive/multiCharacters.zip) the code from GRETA repository *multiCharacters* branch.
10. Go to *{greta-installation}\bin\Player\Lib\External\{your-platform}* and edit the *Plugins_OpenGL.cfg* to replace its relative path to your absolute path.
11. Go to *{greta-installation}\bin* and edit the files *vib.ini* and *Modular.xml* to replace their occurrences of "*empty.xml*" with "*{installation} \UT_HMI\Launchers\couch-environment-for-greta.xml*".
12. Download (http://mary.dfki.de/download/index.html) and install OpenMary TTS and run marytts-component-installer.bat from its bin folder. From that tool, install the following languages: *en-US>cmu-slt*, *en-US>cmu-bdl*and *fr>camille*.

5.3 Execution

1. Run OpenMary TTS server: Open a command line shell, go to *{openmary-installation}\bin* and execute *marytts-server.bat*. Wait until it is up and running. It should run on port 59125.
2. Run GRETA: Open a command line shell, go to *{greta-installation}\bin* and type command java –jar Modular.jar. The GRETA main window will open.
   2.1. Select from the menu: *Add > TTS >OpenMary Client*.
   2.2. Select from the menu*File > Open* and go to *{installation}\UT_HMI\Launchers*, and select *Greta_Modular_Unity-Couch.xml*.
3. Start Docker (if not started before)
4. Open a command line shell, go to *{installation}\UDUN_ArgTech* and type command *docker-compose up*. Wait until it is up and running.
5. Run ASAP from *{installation}\UT_HMI\Launchers\ASAP_Start.bat*. Wait until the message *"Waiting for AgentSpec..."*.
6. Open the Unity project (if not started before). Press play and wait for the agents to take their place.
7. Run Flipper from *{installation}\UT_HMI\Launchers\Flipper_Start.bat*. Wait until the scenario begins. (To restart the dialog you need to restart only Flipper).

5.3.1 Stopping

- Unity: press the play button again (and close the unity editor).
- GRETA: select the shell window with GRETA and press ctrl+c. Do the same with OpenMary.
- ASAP: select the shell window with ASAP and press ctrl+c

- Flipper: select the shell window with Flipper and press ctrl+c
- Docker: select the shell window with the last Docker *docker-compose up*and press ctrl+c, wait for docker containers to quit. (you can then quit Docker from the system tray).

5.3.2 Other configurations

The scenario can be configured according to the presence of the user and its role. This is explained at the end of the readme in GitLab (https://gitlab.com/CouncilOfCoaches/TechnicalDemonstrator).

5.4 Video Material

In order to get a feeling for the status of the Technical Demonstrator, without going through the installation and configuration process defined above, video material is available (and under constant updates on the project's YouTube Channel):

Table 2: Web address for the Council of Coaches YouTube Channel.

https://www.youtube.com/channel/UC8zfyTRCqMsXmGJGLUcrjlQ