Virtual Coaching Activities for Rehabilitation in Elderly

Call: H2020-SC1-2016-2017

Grant Agreement Number: 769807

Deliverable

D5.2 Coaching services interface definition

Deliverable type:	Report
WP number and title:	WP5: Definition of coaching services
Dissemination level:	Public
Due date:	Month 12 – 31 August 2018, revision M16
Lead beneficiary:	AIT
Lead author(s):	Johannes Kropf (AIT)
Co-Author(s):	Lukas Roedl (AIT), Emanuel Sandner (AIT), Karl Kreiner (AIT), Antonio Ascolese (IMA), Vito Nitti (IMA)
Reviewers:	Jordi Rovira (MYS), Stephan Schug (EHTEL)

This project vCare has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 769807.

Document history

Version	Date	Author/Editor	Description
0.1	10.7.2018	Johannes Kropf	Initial version
0.2	8.8.2018	Johannes Kropf	Full list of service descriptions
0.3	14.8.2018	Jordi Rovira	Contributions to section 6 and review
0.4	22.8.2018	Emanuel Sandner, Lukas Roedl	Contribution to section 5 Addressed review comments
0.5	29.8.2018	Johannes Kropf	Updated requirements table Added missing contributions Solved open issues
0.6	30.08.2018	Johannes Kropf, Emanuel Sandner	Final check, ready for review
0.7	31.08.2018	Johannes Kropf, Emanuel Sandner	Addressed review comments from TUD, IMA and EHTEL Finalized deliverable
0.8	07.02.2019	Lukas Roedl	Rework section 4 in relation to the updated architecture in D7.3
0.9	18.02.2019	Lukas Roedl	Rework section 4 and 7 in according to the related deliverables
0.10	28.02.2019	Lukas Roedl	Addition of section 8
0.11	05.03.2019	Antonio Ascolese, Vito Nitti	Added REHABILITY description
0.12	06.03.2019	Lukas Roedl	Rearrange sections and reworked service descriptions
0.13	07.03.2019	Lukas Roedl	Reworked service descriptions and integrated vCaaS descriptions
0.14	08.03.2019	Emanuel Sandner	Fixed References to other deliverables and within the document, updated methodology section, restructured some Sections
0.15	11.03.2019	Lukas Roedl	Updated event descriptions based on input from partners

0.16	11.03.2019	Lukas Roedl	Addressed comments from MYS and TUD
0.17	12.03.2019	Lukas Roedl	Addressed comments from MYS and TUD
0.18	13.03.2019	Lukas Roedl	Addressed comments from MYS and TUD
0.19	14.03.2019	Lukas Roedl	Addressed comments from MYS and TUD
0.20	15.03.2019	Lukas Roedl	Addressed comments from MYS and TUD
0.21	19.03.2019	Emanuel Sandner	Addressed comments
0.22	26.03.2019	Emanuel Sandner Karl Kreiner	Addressed comments Updated “Introduction”, Conclusion and Outlook” and “Executive Summary” Renamed and updated “Events” Section
0.23	27.03.2019	Emanuel Sandner	Added missing Table captions Formatted and finalized deliverable
0.24	01.04.2019	Lukas Roedl Emanuel Sandner	Minor rephrasing in section 4 Updated Service architecture figure

EXECUTIVE SUMMARY

This document presents the outcome of Task 5.2, “Coaching services interface definition”, in which the interfaces for the coaching, data and supporting services, specified in D5.1, are defined. This interface specification is further based on the functional requirements defined in D7.1 and is in line with the common vCare architecture defined in D7.3.

In addition to D5.1, this deliverable describes the coaching, data and supporting service functionalities and interaction models (if applicable) in more detail, by providing implementation details on the used technologies and deployment characteristics within the vCare system. Further, a closer look is given on the service architecture and how it fits in the overall system architecture from D7.3 using a MQTT broker with its publish-subscribe mechanism as medium of information exchange. This approach ensures the service-oriented architecture design principle outlined in D7.1 and D7.3. As some services require information from 3rd-party services, such as weather data, the service design allows the usage of external REST APIs provided by these applications. Similarly, the vCare-as-a-Service concept is discussed here, which provides a REST API for 3rd-party applications. In this case, the REST API component manages the information flow between the internal context bus, realized using MQTT, by exporting a set of functionalities, which can be provided by vCare to other applications. Although the vCare-as-a-Service concept is highly relevant as an exploitable asset for vCare, the information provided here form a conceptual basis and require further work to be streamlined and finalized.

The format of the messages, called “events”, used for communicating with the services is explained in detail along with examples showing how JSON encoded events can look like. These events are categorized in “Event Categories”, in order to give them a clear meaning on what each event is supposed to be used for. The category names are based on the different kind of information that is sent around in the vCare system, such as: “gaming”, “health”, “location”, “reminder” etc., which is derived from the use cases in D1.2, the specified services in D5.1 and from the functional requirements in D7.1. Furthermore, several events with distinct event topics are defined for each event category, covering service requests, results, actions or general messages. Services and other vCare components, like the “Knowledge mediator” from WP4, can subscribe to one or more of those event topics in order to receive them. Likewise, services or other components can publish events with a specific topic to let other interested and subscribed modules know about a new event. The actual information of an event, the payload, is provided by one or more “property” fields, which consist of “key-value” pairs, like e.g. the “game_id” with an id-value for a specific REHABILITY game, a “timestamp” to indicate when an event was created or a more general “content” property which can be used to transmit detailed results of an REHABILITY game session.

For security purposes, a single sign-on solution provided by Keycloak is used for all services and components of vCare. This process allows the usage of tokens following the JSON Web Token specification, which are used for authentication and authorization between the various services and components.

Finally, multi-language support is briefly described and provided by using the commonly used gettext system together with a translation tool like Poedit. This tool enables technical and medical partners to conveniently collaborate and translate all text strings in vCare.

Table of Contents

List of Figures

List of Tables

ACRONYMS

API	Application programming interface
AU	Action Unit
BLE	Bluetooth Low Energy
CS	Coaching Service
DS	Data Service
GUI	Graphical User Interface
HTML	Hypertext Markup Language
i18n	Internationalization
IoT	Internet of things
JSON	JavaScript Object Notation
JWT	JSON Web Token
l10n	Localization
MQTT	Message Queuing Telemetry Transport
PO	Portable Object
POT	Portable Object Template
REST	Representational State Transfer
SS	Supporting Service
UI	User Interface
VC	Virtual coach
vCaaS	vCare-as-a-Service
WP	Work Package
XML	Extensible Markup Language

1 INTRODUCTION

1.1 DOCUMENT SCOPE

Although the title of this document is restricted to coaching services, which are defined as services related to a clinical pathway, data and supporting services are covered here as well. Thus, this document provides the coaching, data and supporting service interface definition and interface description as outcome of Task 5.2 from the list of services specified in D5.1 and from the underlying list of related requirements as given in D5.1 and D7.1. The interfaces defined are in line with the overall architecture as given in D7.3. In addition to the service information provided in D5.1, this deliverable describes also the coaching, data and supporting services functionalities and interaction models (if applicable) in more detail. Furthermore, the concept of the vCare-as-a-Service paradigm is discussed here as well, along with a conceptual specification for a third-party integration. As integral part of the vCare system's functionality, a description on how the identification and authentication mechanism as well as how multi language support is realized is also given in this deliverable.

1.2 METHODOLOGICAL APPROACH

The approach for creating this document, illustrated in Figure 1, is the definition of coaching services from D5.1 along with the functional and non-functional requirements specification from D7.1 and D7.2, and information from the use cases in D1.2.

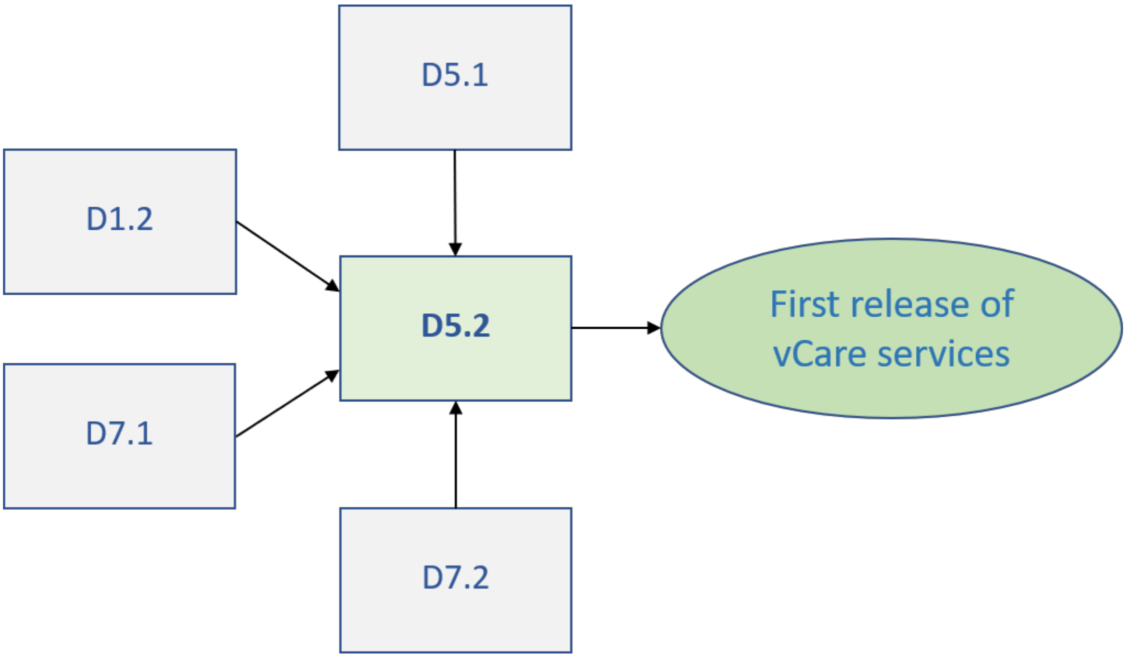


Figure 1: Methodological approach

1.3 DOCUMENT STRUCTURE

The document starts with a description of the overall system architecture related to the services and interfaces (Section 2). Section 3 concentrates on the vCare-as-a-service (vCaaS) concept which enables third-parties to use services provided by vCare. Section 4 describes the communication between the services and the UI Controller. The coaching, supporting and data services and their targeted interactions are described in throughout Section 5, 6 and 7, respectively. The message format along with unique event topics and event categories used for describing service requests, responses and general messages are discussed in Section 8. The sections 9 and 10 describe related topics like authentication and multi-language support. Finally, Section 11 concludes the key aspects of this deliverable and gives an outlook on the further work to be done.

2 SERVICES ARCHITECTURE AND COMMON INTERFACES

Within vCare, the services are realized as software modules running in the cloud. To build a scalable system as stated in D7.2, all communication within the vCare ecosystem is be realized using MQTT[1], while communication with third-party services and APIs is realized using REST[2] interfaces. Figure 2 shows the architecture of the vCare ecosystem considering the services' cloud and its interface to the two directly related vCare layers, which are the *Pathway layer* and the *Knowledge layer*. Components within those layers, like the *Pathway Template Manager*, *Knowledge generator* or *Reasoner* can subscribe to and/or publish information through the MQTT broker, enabling sharing of data between the *Coaching Service layer*, *Pathway layer* and *Knowledge layer*.

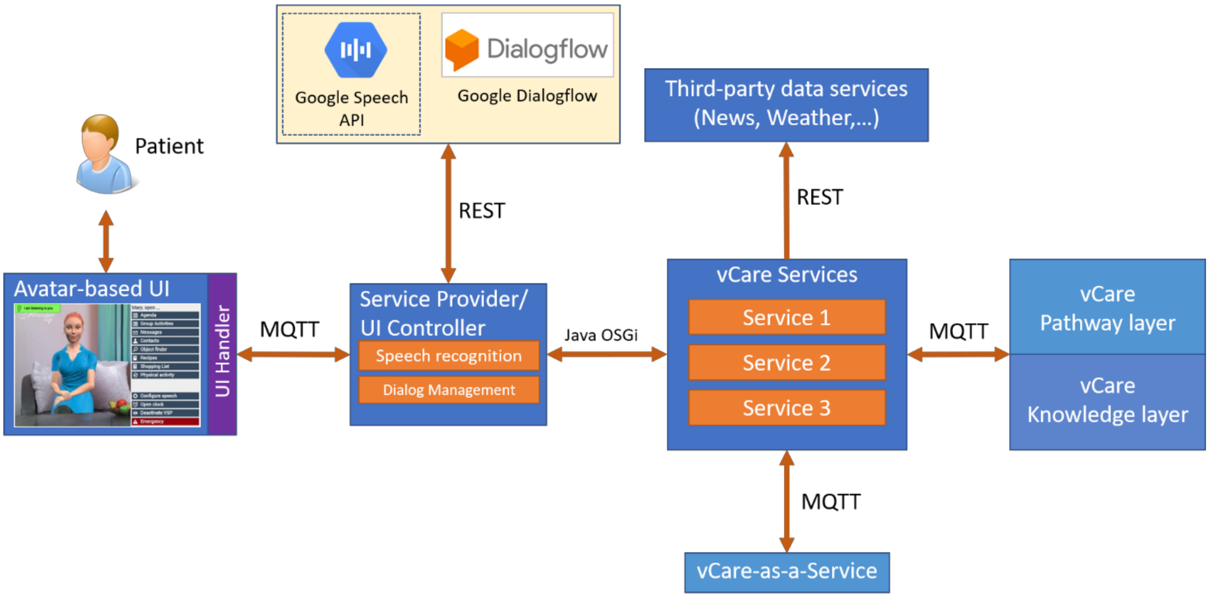


Figure 2: Overall service architecture and connection to other related components

The vCare services' cloud consists of an application server, a MQTT broker and the services, shown in Figure 3. Basically, an application is running on the application server providing functionality to register services (like the ones described in Section 5). These services all implement a common service interface which defines for which dialogue model a service is responsible and enables the interaction with the dialogue model.

A single coaching service is an implementation of the aforementioned interface. This way, the detailed implementation of the service is hidden from the service provider and the attached interfaces. Third-party API's, e.g. for retrieving external data, are only visible for and implemented in the service using it (Figure 3).

Thinking about the integration of third-party applications utilizing the vCare-as-a-service paradigm, an external service can register itself as a possible endpoint and for each of these

[1] Message Queuing Telemetry Transport, https://www.iso.org/standard/69466.html
[2] https://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm

registered services an instance of a proxy service is internally created to handle the communication – see section 3 for details on vCaaS.

The application server also manages the communication to the Google Speech API and the Google Dialogflow Service to enable speech recognition as well as dialogue management.

The UI controller serves as a counterpart for the mobile application respectively the avatar running on the mobile device.

The separate services/modules are interconnected via MQTT with the pathway and knowledge layer from which data is sent and received.

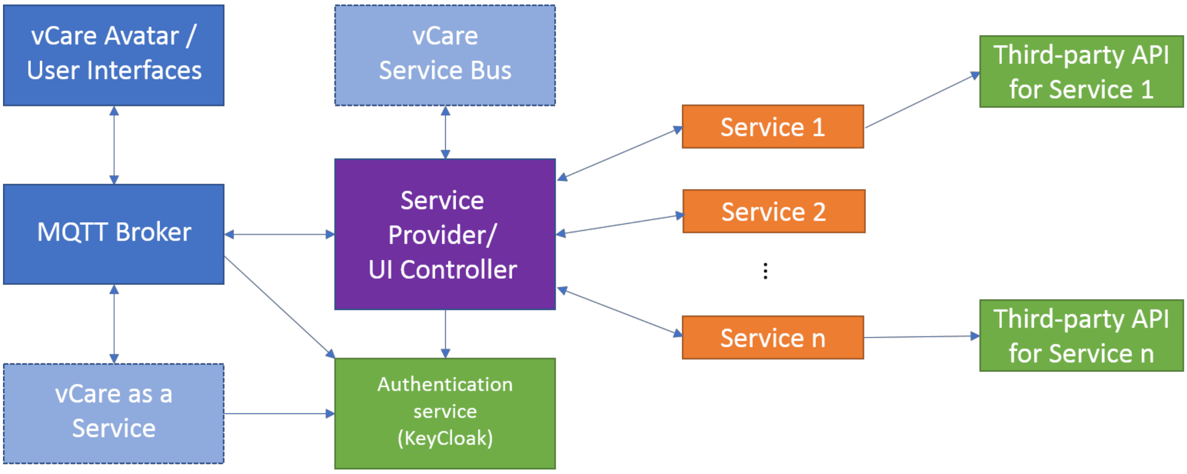

vCare Avatar / User Interfaces
vCare Service Bus
Third-party API for Service 1
Service 1
MQTT Broker
Service Provider/ UI Controller
Service 2
⋮
Service n
Third-party API for Service n
vCare as a Service
Authentication service (KeyCloak)

Figure 3: Coaching service cloud architecture

3 THE VCARE AS A SERVICE PARADIGM

To open the features provided by the vCare system to third-party applications there is the concept of vCare-as-a-Service (vCaaS). Following this concept, vCare will provide services for third party platforms for being able to use vCare services. This way vCare is also opened to new markets, use cases and ecosystems. The vCare exploitation layer will provide appropriate open and standardized interfaces to make this happen. If access in terms of authorization and authentication is granted, access to the vCare system is possible. Hence, in the vCare-as-a-Service paradigm, vCare acts as a content and knowledge provider as well as reasoner for third-party systems providing data to and retrieving from vCare.

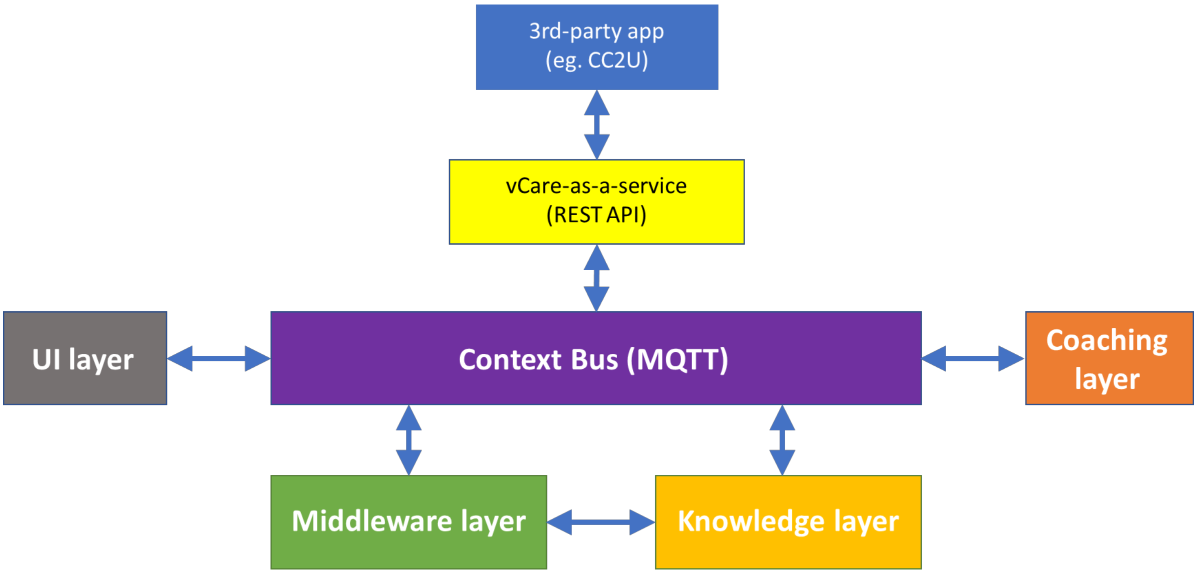


Figure 4: Overall simplified layered architecture with vCare-as-a-service layer

For example, an eHealth platform provider could aim for extending his platform with end-user related services. See section 3.1 for an example on how the integration with the CC2U platform is planned. This is also defined in more details in D7.3. For more detailed information about vCare-as-a-Service also have a look into D7.3 Section 3.3.5.

3.1 CONCEPTUAL SPECIFICATION FOR CC2U INTEGRATION USING VCAAS

The vCare-as-a-Service (vCaaS) is designed to enable the integration of generic services and should also allow the integration with CC2U. CC2U is designed in such a way to allow smooth integration with these services.

As with every other module in the vCare system all of the following API calls pass the authentication and authorization layer where based on policies the access will be granted.

The following logical resources from vCare can be manipulated using API methods:

- Clinical Pathways
- Virtual Coaching Messages
- Agenda/Calendar
- Cognitive Games
- Physical Activity
- Configuration

The vCare API will provide custom methods for querying stats about various entities registered in it. A swagger definition will be provided.

3.1.1 Typical Action Types / Verbs

- **Create, Replace, Update, Delete (CRUD) operations** - GET, POST, PUT and DELETE requests on most logical resources. These are also known as verbs. Please note that not all resources support all the operations because of both technical and business reasons (e.g. some resources should not be deleted because others depend on them; some data is not designed to be returned via GET request etc.).
- **Requesting Data** - There are multiple parameters that, used in conjunction with read (GET) operations, will help query, filter, sort, select, and expand specific information.
- Custom actions and actions which are not tied to a logical resource:
    - Stats methods provide aggregated information on different entities;
    - Account methods provide authentication methods to vCare;

3.1.2 Specific API functionalities

In the following, specific API functionalities (*get, post, put, delete operations*) are presented.

Clinical pathways

The following operations for clinical pathways will be used:

GET Clinical pathways

- Retrieve list of Clinical Pathways
- Retrieve list of Clinical Pathways for specific platform (e.g. CC2U)
- Retrieve list of Clinical Pathways for specific patient
- Retrieve list of Clinical Pathways by Status
- Retrieve Clinical Pathways by name/id

POST Clinical pathways

- Add Clinical Pathways to Patient
- Add Clinical Pathways for specific platform
- Add Clinical Pathways for specific patient

PUT Clinical pathways

- Update Clinical Pathways for specific platform
- Update Clinical Pathways for specific patient
- Update Status of Clinical Pathways (active/obsolete)

DELETE Clinical pathways

- Delete Clinical Pathways for specific platform
- Delete Clinical Pathways for specific patient
- Delete Clinical Pathways by name/id

Virtual coaching

The following operations with regard to virtual coaching will be used:

GET Virtual Coaching Messages

- Retrieve list of Virtual Coaching Messages
- Retrieve list of Virtual Coaching Messages for specific platform
- Retrieve list of Virtual Coaching Messages for specific patient
- Retrieve list of Virtual Coaching Messages by Status
- Retrieve Virtual Coaching Messages by name/id

POST Virtual Coaching Messages

- Add Virtual Coaching Message
- Add Virtual Coaching Message for specific platform
- Add Virtual Coaching Message for specific patients

PUT Virtual Coaching Messages

- Update Virtual Coaching Message for specific platform
- Update Virtual Coaching Message for specific patient
- Update Status of Virtual Coaching Message (sent/not sent)
- Update Status of Virtual Coaching Message with specific name/id

DELETE Virtual Coaching Messages

- Delete Virtual Coaching Message for specific platform
- Delete Virtual Coaching Message for specific patient
- Delete Virtual Coaching Message by name/id

Agenda/Calendar

The following operations with regard to agenda/calendar functions will be used:

GET Security Policy

- Retrieve list of calendar items
- Retrieve list of calendar items for specific platform
- Retrieve list of calendar items for specific patient
- Retrieve list of calendar items by Status
- Retrieve list of calendar items by date
- Retrieve calendar item by name/id

POST Security Policy

- Add calendar item
- Add calendar item for specific platform
- Add calendar item for specific patient

PUT Security Policy

- Update calendar item with specific name/id
- Update Status of calendar item (past/future)

DELETE

- Delete all calendar items for specific platform
- Delete all calendar items for specific patient
- Delete all calendar items with Status=past
- Delete calendar item by name/id

Cognitive games

GET Games

- Retrieve list of Games
- Retrieve list of Games available for specific platform
- Retrieve list of Games for specific patient
- Retrieve results of game by patient
- Retrieve results of game by game name
- Retrieve results of game by result id
- Retrieve statistics by patient

POST Rules

- Add result of game by game name
- Add result of game by patient
- Add result of game by result id

PUT Rules

- Update result of game by game name
- Update result of game by patient
- Update result of game by result id

DELETE

- Delete result of game by game name
- Delete result of game by patient
- Delete result of game by result id

Physical Activity

The following operations with regard to physical activity functions will be used:

GET Physical Activity

- Retrieve list of available physical activity items (steps, calories, movement etc.)
- Retrieve list of available physical activity items for specific platform
- Retrieve list of available physical activity items for specific patient
- Retrieve list of available physical activity items for different filters (start/end date, list of patients, platforms etc.
- Retrieve list of physical activity items by type
- Retrieve physical activity item by type
- Retrieve physical activity item for specific platform
- Retrieve physical activity item for specific patient
- Retrieve physical activity item for different filters (start/end date, list of patients, platforms etc.)

POST Physical Activity

- Add physical activity item by type
- Add physical activity item for specific platform
- Add physical activity item for specific patient

PUT Physical Activity

- Update physical activity item by type

- Update physical activity item for specific platform
- Update physical activity item for specific patient

DELETE Physical Activity

- Delete physical activity item by type
- Delete physical activity item for specific platform
- Delete physical activity item for specific patient

Configuration

The following operations with regard to configuration functions will be used:

GET Config

- Retrieve list of configurations
- Retrieve configuration for specific platform
- Retrieve configuration for specific patient
- Retrieve configuration by type
- Retrieve configuration for different filters

PUT Config

- Update config for specific patient
- Update config for specific platform
- Update config by type

3.2 GENERAL GUIDELINES FOR 3RD PARTY VENDOR INTEGRATION

By allowing products to easily integrate with vCare one can add considerable value to one's product and improve the user's security experience.

As a general guideline for any 3rd party vendors that want to connect with vCare:

1. Build extensive vCare know-how (by properly documenting vCare functionalities and APIs) before starting the testing phase so that the integration problems won't be mistaken as vCare issues.
2. Always verify the communication between the 3rd party vendor product and vCare
3. Always check if the functionalities remain intact following the end-to-end integration process or after a system upgrade (either 3rd party or vCare)

To properly integrate a system with vCare the following methodology should be followed:

1. The proper test infrastructure

   Based on the number of 3rd party interfaces a user can access (mobile, web, desktop) every one of them should be tested with the vCare platform. Also, other variables, should be anticipated and tested for, such as feature functionality across browsers, supported languages, and platform support for third-party integrations.

2. Know-how about vCare platform (API, functionalities that it provides)

Detailed know-how about vCare platform would allow testers to anticipate latent defects related to the integration process, API integrations, and more.

3. Cross-platform check

   A number of platforms will be used to access the 3rd party application, and compatibility testing should be performed to ensure a smooth, intuitive user experience regardless of OS/browser configuration.

4. Perform volume (soak) testing to maintain data integrity

   Typically, a lot of data is uploaded and transferred via the vCare integration. In some cases, a defect in the application can prevent a part of the data from uploading. Soak testing has to be performed to capture these defects before moving the integration to production.

5. Maintain the security of 3rd party users and their data

   Though security is integral in all applications, it is especially relevant for integration with third parties like vCare. Neglecting to test what effect will access of vCare have on a 3rd party user base can have many negative consequences including inconsistencies between user databases, duplicated data entries or missing data.

6. Verify that permissions and restrictions are accurate

   Some of the systems integrated with vCare will have countless users, each with different permission and restriction settings. Possible user scenarios, such as access with read-only/write-only privileges have to be tested and checked to see if they have access to the data provided by vCare

7. Become the end-user

   An effective way to catch defects is to test the integration as if the tester was an actual user of the final system.

4 COMMUNICATION BETWEEN SERVICES AND UI

The communication between the services (cloud), the UI controller (cloud) and the UI handlers (clients) is related to the universAAL UI bus reference architecture[3] on top of MQTT as communication channel. UniversAAL's core feature is the definition of three buses: The *Context Bus*, the *Service Bus* and the *UI Bus*. The *Context Bus* is an event-based communication channel for distributing recognized situation for an unknown recipient (broadcasted) to react on, the *Service Bus* is "call-based", reflecting a classical publish-subscribe paradigm for dedicated services. For a more detailed description have a look to D2.1 Section 4.1. The purpose of the *UI Bus* is to deliver messages related to explicit user interaction. In universAAL, the overall *UI framework* consists of:

- *UI-Handler*: The UI-Handler is responsible of presenting the dialog to the user and capturing the users input. It gets the dialog from and puts the result to the UI bus.
- *UI Bus*: The UI Bus is a middleware bus that works similar to the Service bus; the application sends the dialog, that is to be presented, to the UI Bus which performs a brokerage to find the best way to present the dialog. When the dialog is finished, the result is sent to the UI Bus so that it can be retrieved by the application.
- *Dialog Manager*: The Dialog Manager has mainly two tasks: (1) when a dialog is sent by an application to the UI Bus, it adds adaptation parameters (e.g. the user's location and impairments) and (2) it provides a global menu to search for and invoke services of the system.

In universAAL, the UI is modelled in an abstract way that is independent from device, modality and layout. Such a "Form" includes form controls, which are packaged in three groups:

- *UI Controls* is where common UI Elements are put, including trigger buttons.
- *Submits* are buttons that finish a Dialog and trigger sending the dialogue result back to the UI caller.
- *Standard Buttons* are for system-wide functionality beyond the application logic.

The User Interaction-Handlers (UI-Handlers) are special types of applications in charge of translating these Forms sent by UI Callers to a physical rendering that a user can interact with, such as a GUI, a sound output or Web page. The UI Handler captures the user's response and fills in the information requested by UI Callers into a Form and sends it back to the UI Callers. There can be several UI Handlers in different locations, with different modalities, and the UI Callers are oblivious to them, thus achieving multi-modal and multi-location interaction.

Like defined in the universAAL terminology the UI-Handlers in vCare are running on the client-side and responsible for providing user interaction possibilities to the users including different modalities (e.g. voice, text).

The UI Bus in vCare is realized using the MQTT protocol[4] as communication channel, which is published as standard ISO/IEC 20922:2016. It's a commonly used protocol these days used

[3] https://github.com/universAAL/platform/wiki/UI-Bus-(Details)

[4] Message Queuing Telemetry Transport, https://www.iso.org/standard/69466.html

for IoT applications in terms of message distribution following the publish-subscribe pattern. UI related messages are sent over an MQTT broker, namely Mosquitto[5], which is a performant open source implementation provided by the Eclipse foundation.

On top of the MQTT protocol, an universAAL-like representation of the content using "Forms" is used.

A Dialog is an interaction with a Form. In universAAL, there are four different dialogue types:

- *Main Menu* is for the main screen only and is not used by applications
- *Standard Dialog* is the normal Dialog
- *Subdialog* is a Dialog triggered by a previous one, which comes back after the Subdialog finishes
- *Message* is a popup that may appear regardless of current dialog, on top of it.

In vCare, a similar approach for modelling a user interface and dialogs is used but may be adapted based on specific user interaction or user interface requirements.

Speaking about dialog management the responsible component in vCare is called "UI-Controller". It partly relies on an external cloud service – namely Google Dialog Flow, that already covers major parts of the dialogue management functionality and provides specific API and interfaces for dialogue management. The extension with adaption parameters is done directly by the UI-Controller itself and also prioritization of output to the user is done by that component.

[5] https://mosquitto.org/

5 Description of Coaching Service Interfaces

This section describes each coaching service listed in Table 1 with a description on the general purpose of each service, the targeted interaction between the user and the system (if applicable) and any requirements on personal data (if applicable). The original definition of coaching services can be found in D5.1. For the interactions, the services provide multiple variants of the output which are chosen randomly by the system.

The following sections describe the separate coaching services. In section 8 the underlying events used for communication between the services are described.

Table 1: Coaching Services overview

#	Name	Description	Require-ment	Partner	Implemen-tation	Deploy-ment	Sec-tion
Coaching services							
CS-1	Physical training	Providing motor exercises giving feedback using a depth camera. Additionally, motivational elements help the user staying engaged (rewards, positive feedback etc.).	R5-8	IMA/AIT	Java / Unity / C#	Local device /cloud	5.1
CS-2	Health status	The user is given an overview of his personal health status as well as of the progress of the rehabilitation treatment.	R5-5	AIT/MYS	Java	cloud	5.2
CS-3	Cognitive training	The cognitive training service provides gamified cognitive exercises to the patient. Different levels of difficulty and types of exercises are provided as well as motivational elements through rewards.	R5-9	IMA/AIT	Java / Unity / C#	Local device /cloud	5.3
CS-4	E-learning	Providing general information about health and a healthy lifestyle aiming to increase the patient's health literacy.	R5-2	AIT	Java	cloud	5.4

#	Name	Description	Require-ment	Partner	Implemen-tation	Deploy-ment	Sec-tion
CS-5	Rehabilitation coach	The rehabilitation coach is, on the one hand, providing information about the pathway.	R3-6, R5-3, R5-6, R5-9, R4-9	AIT	Java	cloud	5.5
CS-6	Intelligent notifica-tion/ scheduler	Information of the user about pathway relevant actions using a context-aware intelligent notification service. The notification can be provided via multiple modalities (UI, avatar, speakers, light etc.).	R2-7, R2-8, R3-1, R3-6, R5-2, R5-3, R5-4	AIT/MYS	Java	cloud	5.6
CS-7	Medical questionnaires	Based on validated and standardized questionnaires, data related to the health status of the patient shall be collected to support the monitoring of health status progression as well as the diagnosis of co-morbidities.	R5-7	AIT	Android/ Web technologies	Local/cloud	5.7
CS-8	User feeling	The avatar asks the user about his or her subjective health status (physically and mentally). This is defined in the personalized pathway.	R5-6	AIT	Java	cloud	5.8
CS-9	Speech and swallow-ing therapy	vCare shall provide digital support for therapy of dysphonia, dysarthria, oropharyngeal dysphagia via serious games.	R5-10	AIT	Java/Unity	local	5.9

5.1 CS-1: Physical Training

Description

Rehabilitation system based on motion tracking through 3D depth sensors, providing game-based personalized exercises with immediate feedback to the user and data collection for professionals. The serious games are triggered either by the user or by the avatar via a reminder. The physical exercise is then invoked automatically.

A specific game plan is available after configuration on the caregiver-interface and can be retrieved through the event topics defined in section 8.1.

Targeted interaction to start the physical exercises 1

System: “You haven’t performed your physical exercise for today yet. Do you want to do the exercises right now?”

User: “Yes, please start”.

System: “Ok, I will switch on the TV and start your physical exercise program.”

Targeted interaction 2

User: “Please start my physical training” | “I would like to start my physical training now”

System: “Ok, I will start the physical training program for you, please take your place in front of the TV”.

Requirements on personal data

- User id
- Physical therapy plan

5.2 CS-2: Health status

Description

The user shall be informed about his/her current health status and progress in the individual rehabilitation pathway. Aggregated health data are presented to the user in form of numbers, graphs and speech output. In addition, deviations from the normal values are also shown. The deviations are determined in an underlying service within the knowledge layer.

Targeted interaction 1

User: “Show me my health status” | “Give me my health data” | “what are my last measurements?”

System: “Your blood pressure is normal today, your physical activity index increased by 3% during the last week”.

Requirements on personal data

- User id
- Summary of health status (vital parameters trend)

5.3 CS-3: Cognitive Training

Description

The user can play cognitive games, initiated by communication with the avatar. Cognitive games are third-party apps provided by IMA and are part of the REHABILITY software. The games themselves are not integrated into the UI controller, but are running in a separate app. The games selection, levels, etc. are configured by a care professional.

Targeted interaction 1

User: “Please start the cognitive games”

System: launches cognitive games

Targeted interaction 2

System: “Today cognitive training is in your agenda; shall I launch cognitive games for you?”

User: “Yes, please”

System: launches cognitive games

Targeted interaction 3

System: “Today cognitive training is on your agenda; shall I launch cognitive games for you?”

User: “No, later please”

System: postpones the request and asks again later. (“later” is derived from the time to user usually performs the cognitive training)

Requirements on personal data

- User Id
- Cognitive rehabilitation plan

5.4 CS-4: E-LEARNING

Description

To increase the patient's health literacy, e-learning components related to a healthy lifestyle and the user's rehabilitation pathway are provided to the user. The content is provided by the avatar in subtle way, e.g. in form of daily tips.

Targeted interaction 1

System (avatar): "Did you know that smoking increases the arterial stiffness and the blood pressure?"

Requirements on personal data

- User id
- Lifestyle habits
- Rehabilitation pathway

5.5 CS-5: REHABILITATION COACH

This service aims for assisting the patient through his/her rehabilitation process by providing recommendations for a lifestyle which is supporting the rehabilitation plan. The detailed information to be provided is defined in the patient's personal pathway.

Targeted interaction 1:

System states: "The weather is fine today, I recommend you going for a walk outside. You didn't do for a while."

User can respond: "Ok, thank you."

Targeted interaction 2:

System states: "Your blood pressure was a bit high during the last weeks. Please be aware of your salt intake".

User can respond: "Ok, thank you."

Requirements on personal data

- User id
- Frequency the user shall be informed

5.6 CS-6: Intelligent notifications/scheduler

Description

This service provides a mechanism to notify the user about outstanding tasks or relevant information. Additionally, this service informs the patient about his daily schedule according to the rehabilitation plan, which can be configured via the caregiver user interface and is part of the KIOLA platform[6], developed and provided by AIT (for an example see Figure 5).

Notifications can be triggered by any service or module within the vCare ecosystem via MQTT. The reminder and notification service ensure that the user retrieves the information and the user is able to postpone it. Context information is used for efficient triggering.

Targeted interaction 1

System: “I have a new notification for you: Please do not forget to take your pills.”

User: “Thank you, please remind me again later.”

Targeted interaction 2

System: “You have just measured your blood pressure and the current value is very high. Please take some rest or follow the instructions given by your doctor. I will remind you to do an extra measurement after a few hours”.

User: “Ok, thanks”

Targeted interaction 3

User: “What is my schedule for today?” | “What is on the agenda today?” | “Do I have anything on the agenda today / tomorrow?”

[6] M. Drobics, K. Kreiner, H. Leopold: "*Next Generation ICT Platform to Harmonize Medical, Care and Lifestyle Services*"; in: "*Advances in Intelligent Systems and Computing*", 399; Springer, 2016, ISBN: 978-3-319-25731-0, S. 275 - 283.

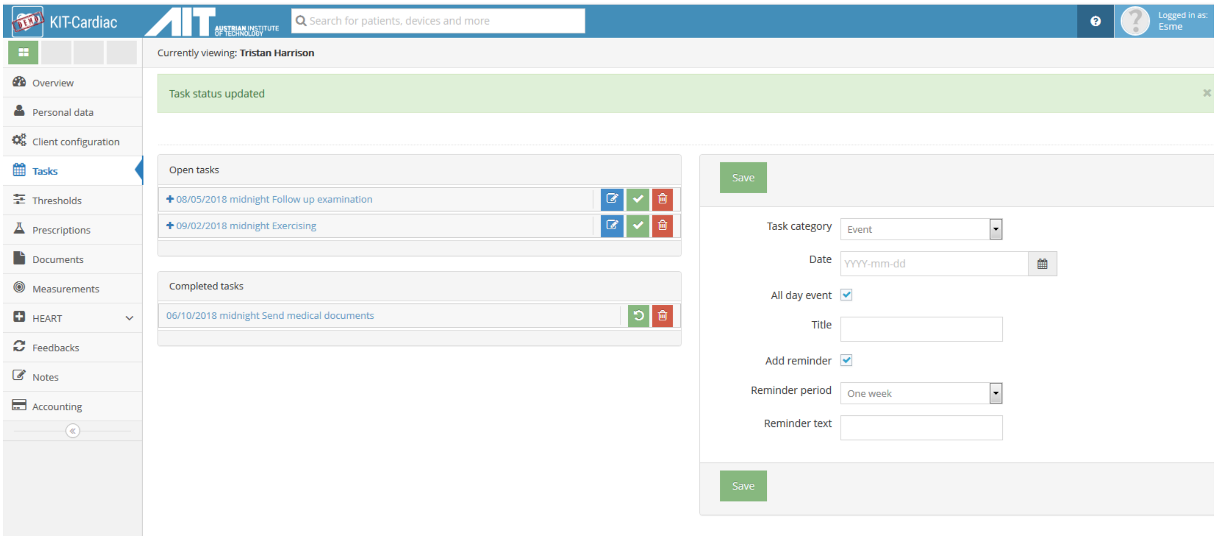
System response: “Today there is one unit of physical exercise planned. Shall I start the training now?”

Targeted interaction 3:

System states: “The last week you haven’t been very active. I suggest you to do more physical exercises, maybe you can go out for a one hour walk today.

Requirements on personal data

- User id
- Personal lifestyle recommendation



Currently viewing: **Tristan Harrison**

Task status updated

Open tasks

08/05/2018 midnight Follow up examination

09/02/2018 midnight Exercising

Completed tasks

06/10/2018 midnight Send medical documents

Task category: Event

Date: YYYY-mm-dd

All day event ☑

Title:

Add reminder ☑

Reminder period: One week

Reminder text:

Figure 5: Screenshot of the agenda manager in the caregiver UI

5.7 CS-7: Medical Questionnaires

Description

This service provides a way to perform standardized medical questionnaires at the user’s home via a tablet interface. This is on top of the avatar and provided via touch input. One example for the questionnaire is the HADS for anxiety and depression scale. The answers to the questionnaire are stored and used to support the monitoring of the health status progression as well as the diagnosis of co-morbidities.

This service is system triggered, that means the system initiates the interaction and is scheduled according to the patient’s rehabilitation plan.

Targeted interaction:

System: “I have a HADS questionnaire for you. This will help me to keep track of your health status. Please take some minutes to fill it out now.”

User: “Ok, show me the questionnaire.”

The system shows the questionnaire and collects the patient’s answers.

Requirements on personal data

- User id
- Rehabilitation pathway

5.8 CS-8: USER FEELING

This service aims to retrieve the user feeling by asking the user with a frequency as defined by the clinician, e.g. once a day. The user’s response is classified and stored in the common database.

This service is system triggered, that means the system initiates the interaction.

Targeted interaction:

System asks: “How do you feel today?” | “How are you today?”

User can respond: “Good” / “Dizzy” / “Weak” / “Bad” / “Not good“ / “Tired”

Requirements on personal data

- User id
- Frequency the user shall be asked

5.9 CS-9: SPEECH AND SWALLOWING THERAPY

This service aims to provide digital support for therapy of dysphonia, dysarthria, oropharyngeal dysphagia via the avatar. The avatar based virtual coach guides the patient through breathing exercises following a pre-defined therapy plan. Avatar gestures support the exercises.

This service is system triggered, that means the system initiates the interaction.

Targeted interaction 1:

System asks: “Today it is time for your speech therapy. Shall I start the exercises now?”

User can respond: “No”

System responses: “Ok, I will come back to you later.”

Targeted interaction 2:

System asks: “Today it is time for your speech therapy. Shall I start the exercises now?”

User can respond: “Yes, please | Yes”

System responses: “Ok, please stand came and in upright position in front of me. We start in 5 Seconds.”

System states: “We will start with a 5-cycle inspiration exercise with a 5 second pause. Breath-in – (5 secs pause) - breath out - (5 secs pause) - breath in - (5 secs pause) - breath out – (5 secs pause) - breath in - (5 secs pause) – breath out - (5 secs pause) - breath in – (5 secs pause) - breath out - breath in – breath out.

System states: “Perfect, we continue with a slow expiration exercise. Breath in – (4 secs pause) – breath out slowly; 4, 3, 2, 1. Great.”

System states: “Perfect, we continue with a slow expiration exercise. Breath in – (4 secs pause) – breath out slowly; 4, 3, 2, 1. Great. Let’s continue.”

System states: “Breath in – (3 secs pause) – “breath out slowly - 3 - 2 - 1. Perfect!”

Systems states: “Now breath by speaking the letter s. Breath in through your nose and hold the air for 3 seconds. Then breath out slowly and feel how your belly gets in- and deflated. Let’s start. Breath in - 3 - 2 - 1, hold for 3 - 2 - 1 seconds, now slowly breath out - 3 - 2 - 1.”

Systems states: “Ok, one more time, but now spell ‘a’. Breath in through your nose and hold the air for 3 seconds. Then breath out slowly and feel how your belly gets in- and deflated. Let’s start. Breath in - 3 - 2 - 1, hold for 3 - 2 -1 seconds, now slowly breath out - 3 - 2 - 1.”

System states: “That’s it for today, how did you feel?”

User can respond: “Good | hard | bad | fine”.

The system responds accordingly, e.g. “Good to hear that | Will be better next time”.

Requirements on personal data

- User id
- Frequency of the exercise to be performed.
- Concrete therapy plan (repetitions, speed).

6 DESCRIPTION OF SUPPORTING SERVICE INTERFACES

This section describes each supporting service listed in Table 2 with a brief description on the general purpose of each service, at least one sample interaction between the user and the system (if applicable) and any requirements on personal data (if applicable). Like with Table 1 the mentioned services are defined in D5.1 and listed here to give a deeper insight on how the interfaces provided/used by these services look like and where they are implemented.

Table 2: Supporting Services overview

#	Name	Description	Require-ment	Partner	Implemen-tation	Deploy-ment	Section
Supporting Services							
SS-1	Weather	Patient can ask the system for the current weather and the forecast.	R3-10	AIT	Java	cloud	6.1
SS-2	Agenda	Patient can ask for his non-pathway related agenda.	NA (see SS-2 related use cases in D5.1)	AIT	Java	cloud	6.2
SS-3	Standby	The Patient can put the VC in a standby/passive mode to not be disturbed-	R3-11	AIT	Java	Tablet /cloud	6.3

6.1 SS-1: WEATHER

Description

The User can ask the system for the weather forecast of the current day and/or the next day for a specific location. If no location is given, the current location as defined in the user profile is used. For retrieving the data, a free and open third-party solution is used: http://www.openweathermap.org

The service provides a REST API providing data in JSON, XML or HTML format, e.g. the request api.openweathermap.org/data/2.5/weather?lat=35&lon=139 provides

```
{"coord":{"lon":139,"lat":35},
"sys":{"country":"JP","sunrise":1369769524,"sunset":1369821049},
"weather":[{"id":804,"main":"clouds","description":"overcast clouds","icon":"04n"}],
"main":{"temp":289.5,"humidity":89,"pressure":1013,"temp_min":287.04,"temp_max":292.04},
"wind":{"speed":7.31,"deg":187.002},
"rain":{"3h":0},
"clouds":{"all":92},
"dt":1369824698,
"id":1851632,
"name":"Vienna",
"cod":200}
```

The weather service converts the weather data in human understandable text in the language used as base for speech output.

Targeted interaction 1

User: “How is the weather?” / “How is the weather in Vienna today?”

System response: “The weather today in Vienna will be sunny. The temperature ranges from 20 to 25 degrees.”

Requirements on personal data

- Location
- Language

6.2 SS-2: AGENDA

Description

This service provides information and reminders about the personal, non-pathway related agenda of the patient, like having an appointment with a friend/family or at a hairdresser.

Targeted Interaction 1

User asks: “What is my personal agenda for today?” | “Do I have any personal appointments today?”

System responds: “Yes, you have an appointment with John today at 11am”

The system additionally shows the appointment with further information (e.g. location) on the tablet screen.

Targeted Interaction 2

System: “You have an appointment at the hairdresser scheduled in one hour at 1pm.”

User responds: “Ok, thank you.”

Requirements on personal data

- User id
- Frequency how often the patient shall be reminded

6.3 SS-3: Standby

Description

The service enables the user to put the system in standby mode by voice input and/or button click (touch). In standby, the voice detection is switched off as well as there is no speech output provided by the system. In this mode, notifications are shown only on the display in a non-obtrusive way indicating that there are notifications to be shown.

The avatar indicates that the system is not active / listening, e.g. the avatar sits down on the couch or leaves the visible area. Both animations are provided to make the avatar more natural (not repeating the same activities).

Targeted Interaction 1

User asks: “Go to sleep” / “go to standby” / “be quiet” / “switch on standby”

System responds: “I’m now in standby, please let me now when you need me again”

Targeted Interaction 2

User pushes button “leave standby” or “wakeup”

System responds: “What can I do for you?”

Requirements on personal data

- None

7 DESCRIPTION OF DATA SERVICE INTERFACES

This section describes each data service listed in Table 3 with a brief description on the general purpose of each service, at least one sample interaction between the user and the system (if applicable). Like with Table 1 the mentioned services are defined in D5.1 and listed here to give a deeper insight on how the interfaces provided/used by these services look like and where they are implemented.

Table 3: Data Services overview

#	Name	Description	Require-ment	Part-ner	Implemen-tation	Deploy-ment	Sec-tion
Data services							
DS-1	Location of the patient in the house and home activity/ inactivity monitoring	Detection of the location of the end-user on room level and measurement of time where the user spends most of the time at home. This service also detects when the user leaves the house.	R2-1, R2-6, R2-7	MYS	Java / Android	Tablet/ Cloud	7.1
DS-2	Body position detection	Detection of the current body pose of the patient and relies on the use of a wearable sensor (Angel4). The data analysis is performed off-device.	R2-3	MYS	Java / Android	Tablet/ Cloud	7.2
DS-3	Object use detection	Detection if certain objects are used. This is realized by mounting tag sensors on objects, like the cane.	R2-4	MYS	Java / Android	Tablet/ Cloud	7.3
DS-4	Fall detection	Detection of falls using a wearable device (Angel4)	R2-5	MYS	Java / Android	Tablet/ Cloud	7.4
DS-5	Monitoring of vital parameters	Collecting vital parameters using personal health devices (blood pressure monitor,	R7-1, R7-2, R7-3, R7-4, R7-5,	SIV	Java / Android	Tablet/ Cloud	7.5

#	Name	Description	Require-ment	Part-ner	Implemen-tation	Deploy-ment	Sec-tion
		balance board, weight scale …	R7-6, R7-7				
DS-6	Monitoring of physical activity level	The monitoring of physical activity includes used energy, number of steps, pace etc.	R2-6	MYS	Java / Android	Tablet/ Cloud	7.6
DS-7	Detection of activities of daily living	Collection of the user's behaviour at home, e.g. time at the bed, time watching TV, the time consumed for lunch etc. This services also detects deviation from expected behaviours.	R2-8, R2-9	MYS	Java / Android	Tablet/ Cloud	7.7
DS-8	Social monitoring	This service detects if there are visitors at home and records the patient going out. This service is on top of DS-1 using the same sensor data.	R2-10	MYS	Java / Android	Tablet/ Cloud	7.8
DS-9	Detection of emotional state	The emotional state of the user is estimated based on voice and facial expressions when communicating with the virtual coach.	R3-3	AIT	Java / Android	Tablet/ Cloud	7.9

7.1 DS-1: LOCATION OF THE PATIENT IN THE HOUSE AND HOME ACTIVITY/INACTIVITY MONITORING

Description

The service aims at localizing the current location of the user within the house on room level. Transitions from one room to another are recorded as state changes. To detect the current location Bluetooth LE based presence detection sensors are used. The data is gathered by the tablet making some local post-processing tasks to filter relevant location patterns. Once this process is done, the information is transmitted through the specific IoT agent to the

FIWARE broker in the platform, which makes it available to the knowledge layer. The current location is also used by the UI controller as context relevant information. The state change is published to listening services via MQTT.

The overall smart home / client-side architecture is described in detail in D2.1. A functional overview is depicted in Figure 6. The data transmission across the vCare platform is represented in Figure 7. Basically, the communication is based on the standardized protocols and interfaces MQTT and REST, the data itself are structured in JSON.

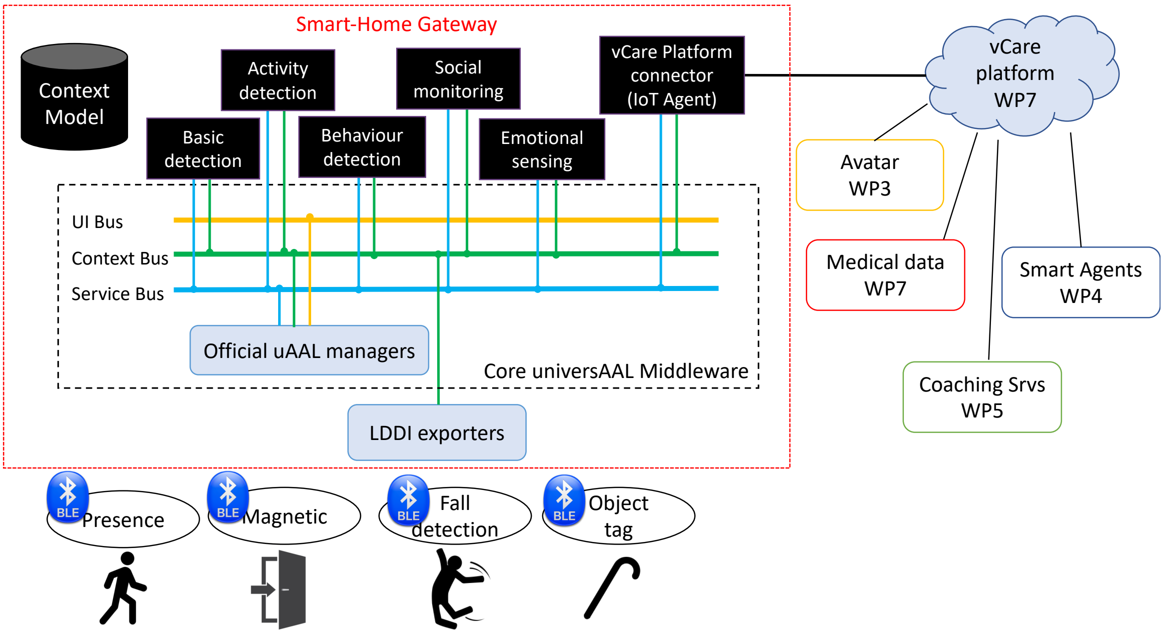


Figure 6: vCare Smart home functional overview (from D2.1)

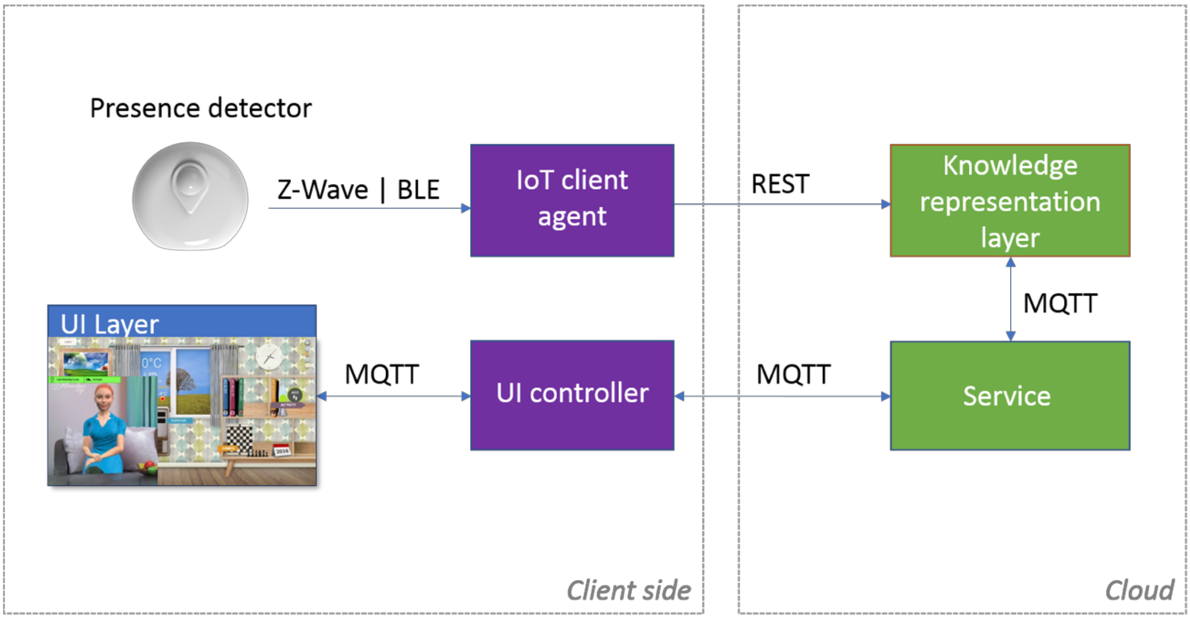


Figure 7: Data interfaces for smart home and context related data

7.2 DS-2: BODY POSITION DETECTION

Description

This service is using a waist wearable device (activity tracker) to estimate the user's current pose (sitting, standing, walking, lying). The same wearable provides other information regarding step counting, falls and motor symptoms for people with Parkinson Disease.

7.3 DS-3: OBJECT USE DETECTION

Description

This service is collecting the distance of supporting objects like canes that are important to reduce the risk of falling. A Bluetooth LE tag is fixed in the key elements to observe the risk habits of the patient and build her risk profile.

7.4 DS-4: FALL DETECTION

Description

Detection of falls and stand-up events using a wearable device (Angel4), if available.

7.5 DS-5: MONITORING OF VITAL PARAMETERS

Description

This service aims at collecting vital parameters from the end-user by personal health devices. Regarding to the requirements, these are the elements prioritized in D7.2:

- Blood pressure monitor
- Weighing scale
- Activity counter (Step counter)
- Heart rate monitor

The personal health devices are connected to the tablet via Bluetooth Low Energy (BLE), which is acting as a gateway and transmits the data to the vCare knowledge representation layer. Notifications and/or reminders (CS-6: Intelligent notifications/scheduler) are used to notify the user about the necessity to use the personal health device in the case of not continuous monitoring (e.g. weighing scale, blood pressure monitor). Some devices allow continuous measurement in the background without user interaction.

7.6 DS-6: MONITORING OF PHYSICAL ACTIVITY LEVEL

Description

A smart watch (Fitbit Pure Pulse) measures physical activity continuously in- and outdoors and transmits the data to the vCare knowledge representation layer. It is aimed that the smart watch transmits data as soon as it is in reach of the gateway (tablet via Bluetooth Low Energy) in the background without any interaction with the user.

7.7 DS-7: Detection of activities of daily living

Description

This service is strongly related to DS-1 and is based on the same sensors among additional ones. It aims at the detection of basic behaviours like:

- Inactivity
- Sleeping
- Leaving the home
- Coming home
- Cooking

This service does not include any interaction with the end-user put provides data for the knowledge base to be analysed. For more information, refer to D2.2 section 4.6.

7.8 DS-8: Social monitoring

Description

This service detects if there are visitors at home and records the patient going out. This service is on top of DS-1 using the same sensor data.

7.9 DS-9: Detection of emotional state

Description

This service estimates the emotional state of the patient based on voice analysis and facial expressions when communicating with the VC. The reader is advised to consult D3.1 for a detailed description of Emotion Recognition and how the emotion is recognized from voice or facial expressions.

8 EVENTS

This section is about how the separate services can communicate with each other and the UI controller. The communication for this kind of information is MQTT, as described previously. The format of the messages, here we call them “events”, is described on the following pages.

The in-depth technical description of each service can be found in the D5.3, when released.

The vCare Coaching Services are technically realized using a Java framework based on-top of the OSGi[7] technology, a standard which enables modularization for Java applications. In this OSGi framework, most of the services are implemented, one important concept is the concept of sending and receiving (a)synchronous events between the modules for communication purposes. This works in the framework internally with features provides by the framework itself and for the communication with services not directly running in the framework these events are sent over MQTT. The tables on the following pages describe the event types and the event properties defined for the vCare services. Furthermore, examples are given in JSON format, how these events can look like.

The separate event types are clustered in the categories:

- Event Category "GAMING"
- Event Category "HEALTH"
- Event Category "LOCATION"
- Event Category "MESSAGE"
- Event Category "POSITION"
- Event Category "REMINDER"
- Event Category "SUPPORT"

Each event has a related topic which specifies the type of an event. All topics defined in vCare start with `eu/vcare/event/` followed by a category identifier. An event also can have one or more properties set. A property consists basically of a key and a corresponding value. All the specified properties for a certain topic are also included in the descriptions in the following sections.

8.1 EVENT CATEGORY "GAMING"

Event topics linked to this category are all related to gaming and are shown in Table 4.

Table 4: Event Topics for Event Category "GAMING"

GAME_RESULT	A message sent from the gaming module
GAME_SESSION_PLANNED_REQUEST	A message sent to the UI professional module
GAME_SESSION_PLANNED_RESULT	A message sent from the UI professional module
GAME_SESSION_PLAYED_REQUEST	A message sent to the gaming module
GAME_SESSION_PLAYED_RESULT	A message sent from the gaming module

[7] https://www.osgi.org/

GAME_SESSION_START	A message sent to the gaming module
GAME_SESSION_STARTED	A message sent from the gaming module
GAME_SESSION_COMPLETED	A message sent from the gaming module

8.1.1 GAME_RESULT

This is a message that is sent from the gaming module to indicate a specific game was finished. Table 5 shows the details and Table 6 shows the properties of this event topic.

Table 5: Details of Event Topic "GAME_RESULT"

Name	GAME_RESULT
Topic	eu/vcare/event/gaming/game/result
Use	A message sent from the gaming module
References	CS-1, CS-3

Table 6: Properties of Event Topic "GAME_RESULT"

Property	Class	Description
CONTENT [2]	JsonElement	A JSON object describing the content
GAME_ID [1]	String	A string describing the game id
INTERACTION_ID	String	A string specifying an interaction id to match responses to requests
SESSION_TYPE	GameSessionType	An enum describing the game session type
SESSION_ID [1]	String	A string describing the game session id
TIMESTAMP	Long	A timestamp representing the number of seconds since 01.01.1970 00:00:00 UTC
USER_ID	String	A string specifying a user id

[1] property is optional

[2] value of the property can be *null*

GAME_RESULT example.

```
{
  "topic": "eu/vcare/event/gaming/game/result",
  "properties": {
    "INTERACTION_ID": "155742af-4942-4142-8b81-184d2edd1393",
    "TIMESTAMP": 1552320821447,
    "USER_ID": "123456",
    "SESSION_TYPE": "cognitive",
    "GAME_ID": "135",
    "CONTENT": {
      "pg_id": "Plan game id as obtained from the loading of the plan",
      "session_id": "Actual session (Plan instance) ID",
      "status": "Can be \u0027complete\u0027, \u0027fail\u0027 or
\u0027userexit\u0027 to identify whether the user complete successfully the game,
fails or just exit without finish",
      "score": "The score obtained during the game",
      "results": {
        "description": "A custom JSON object with everything the developer want to
track about the game"
      },
      "tracking": {
        "description": "Optional, a JSON object containing the skeleton data
tracked at sampling rate"
      }
    }
  }
}
```

The *tracking* field shall contain:

A list of points in space at the given sampling rate e.g. *"right-hand": [[x, y, z], …]* for specific joints like right-hand, left-hand, torso, right ankle or left-ankle there is also the possibility to have each frame a json specifying other properties like

```
{
    p: [x, y, z], # same as before
    s: 100, # the score obtained in that frame
    g: 1, # for left-hand and right hand identifies the grab gesture (c for the
click)
}
```

Other available (mandatory) joints are *left-shoulder*, *head*, *left-hand*, *right-knee*, *right-ankle*, *right-elbow*, *collar*, *right-hip*, *left-ankle*, *left-elbow*, *torso*, *right-shoulder*, *waist*, *left-knee*, *right-wrist*, *left-wrist*, *neck* and *left-hip*

- *root*: The *root* node identifies the starting position of the user in device coordinates e.g. *[26, -1220, 2402]*
- *f*: The node *f* describes the sampling frequency e.g. 20

8.1.2 GAME_SESSION_PLANNED_REQUEST

This is a message that is sent to the UI professional module to get a list of planned sessions related to a specific user. Table 7 shows the details and Table 8 shows the properties of this event topic.

Table 7: Details of Event Topic "GAME_SESSION_PLANNED_REQUEST"

Name	**GAME_SESSION_PLANNED_REQUEST**

Topic	eu/vcare/event/gaming/session/planned/request
Use	A message sent to the UI professional module
References	CS-1, CS-3

Table 8: Properties of Event Topic "GAME_SESSION_PLANNED_REQUEST"

Property	**Class**	**Description**
CONTENT [2]	JsonElement	A JSON object describing the content
INTERACTION_ID	String	A string specifying an interaction id to match responses to requests
SESSION_TYPE	GameSessionType	An enum describing the game session type
TIMESTAMP	Long	A timestamp representing the number of seconds since 01.01.1970 00:00:00 UTC
USER_ID	String	A string specifying a user id

[2] value of the property can be `null`

GAME_SESSION_PLANNED_REQUEST example.

```
{
  "topic": "eu/vcare/event/gaming/session/planned/request",
  "properties": {
    "INTERACTION_ID": "155742af-4942-4142-8b81-184d2edd1393",
    "TIMESTAMP": 1552320821447,
    "USER_ID": "123456",
    "SESSION_TYPE": "cognitive",
    "CONTENT": {
      "filter_timestamp_start": 1551321273119,
      "filter_timestamp_end": 1551369273511
    }
  }
}
```

8.1.3 GAME_SESSION_PLANNED_RESULT

This is a message that is sent from the UI professional module as a response to a previous request related to a specific user. Table 9 shows the details and Table 10 shows the properties of this event topic.

Table 9: Details of Event Topic "GAME_SESSION_PLANNED_RESULT"

Name	**GAME_SESSION_PLANNED_RESULT**
Topic	eu/vcare/event/gaming/session/planned/result
Use	A message sent from the UI professional module

References	CS-1, CS-3

Table 10: Properties of Event Topic “GAME_SESSION_PLANNED_RESULT”

Property	Class	Description
CONTENT [2]	JsonElement	A JSON object describing the content
INTERACTION_ID	String	A string specifying an interaction id to match responses to requests
SESSION_TYPE	GameSessionType	A enum describing the game session type
TIMESTAMP	Long	A timestamp representing the number of seconds since 01.01.1970 00:00:00 UTC
USER_ID	String	A string specifying a user id

[2] value of the property can be *null*

GAME_SESSION_PLANNED_RESULT example.

```
{
  "topic": "eu/vcare/event/gaming/session/planned/result",
  "properties": {
    "INTERACTION_ID": "155742af-4942-4142-8b81-184d2edd1393",
    "TIMESTAMP": 1552320821457,
    "USER_ID": "123456",
    "SESSION_TYPE": "cognitive",
    "CONTENT": {
      "sessions_planned": [
        {
          "session_id": "The session ID",
          "total": "Total count of games in the session",
          "games": [
            {
              "Plan_Game_ID": "Plan game id that identifies the game in the
template plan",
              "Game_ID": "ID of the game type",
              "Name": "Title of the game"
            }
          ],
          "schedule": "JSON object describing the specified plan for this session"
        }
      ]
    }
  }
}
```

8.1.4 GAME_SESSION_PLAYED_REQUEST

This is a message that is sent to the gaming module to get a list of played sessions related to a specific user. Table 11 shows the details and Table 12 shows the properties of this event topic.

Table 11: Details of Event Topic "GAME_SESSION_PLAYED_REQUEST"

Name	GAME_SESSION_PLAYED_REQUEST
Topic	eu/vcare/event/gaming/session/played/request
Use	A message sent to the gaming module
References	CS-1, CS-3

Table 12: Properties of Event Topic "GAME_SESSION_PLAYED_REQUEST"

Property	Class	Description
CONTENT [2]	JsonElement	A JSON object describing the content
INTERACTION_ID	String	A string specifying an interaction id to match responses to requests
SESSION_TYPE	GameSessionType	An enum describing the game session type
TIMESTAMP	Long	A timestamp representing the number of seconds since 01.01.1970 00:00:00 UTC
USER_ID	String	A string specifying a user id

[2] value of the property can be *null*

GAME_SESSION_PLAYED_REQUEST example.

```
{
  "topic": "eu/vcare/event/gaming/session/played/request",
  "properties": {
    "INTERACTION_ID": "155742af-4942-4142-8b81-184d2edd1393",
    "TIMESTAMP": 1552320821457,
    "USER_ID": "123456",
    "SESSION_TYPE": "cognitive",
    "CONTENT": {
      "filter_timestamp_start": 1551321273119,
      "filter_timestamp_end": 1551369273511
    }
  }
}
```

8.1.5 GAME_SESSION_PLAYED_RESULT

This is a message that is sent from the gaming module as a response to a previous request related to a specific user. Table 13 shows the details and Table 14 shows the properties of this event topic.

Table 13: Details of Event Topic "GAME_SESSION_PLAYED_RESULT"

Name	**GAME_SESSION_PLAYED_RESULT**
Topic	eu/vcare/event/gaming/session/played/result
Use	A message sent from the gaming module
References	CS-1, CS-3

Table 14: Properties of Event Topic "GAME_SESSION_PLAYED_RESULT"

Property	**Class**	**Description**
CONTENT [2]	JsonElement	A JSON object describing the content
INTERACTION_ID	String	A string specifying an interaction id to match responses to requests
SESSION_TYPE	GameSessionType	An enum describing the game session type
TIMESTAMP	Long	A timestamp representing the number of seconds since 01.01.1970 00:00:00 UTC
USER_ID	String	A string specifying a user id

[2] value of the property can be *null*

GAME_SESSION_PLAYED_RESULT example.

```
{
  "topic": "eu/vcare/event/gaming/session/played/result",
  "properties": {
    "INTERACTION_ID": "155742af-4942-4142-8b81-184d2edd1393",
    "TIMESTAMP": 1552320821457,
    "USER_ID": "123456",
    "SESSION_TYPE": "cognitive",
    "CONTENT": {
      "sessions_played": [
        {
          "session_id": "The session ID",
          "total": "Total count of games in the session",
          "results": [
            {
              "Plan_Game_ID": "Plan game id that identifies the game in the
template plan",
              "Status": "Can be toplay, complete or userexit identifies the final
status of the game",
              "Game_Start": "Start timestamp of the game (when patient click play
button after the briefing)",
              "Game_End": "End timestamp just when all the time elapsed",
              "Score": "Final score of the play",
              "Results": "JSON object containing the whole results of the play its
different for every game",
              "Game_ID": "ID of the game type",
              "Name": "Title of the game"
            }
          ]
        }
      ]
    }
  }
}
```

8.1.6 GAME_SESSION_START

This is a message that is sent to the gaming module to start a specific session of games. Table 15 shows the details and Table 16 shows the properties of this event topic.

Table 15: Details of Event Topic "GAME_SESSION_START"

Name	**GAME_SESSION_START**
Topic	eu/vcare/event/gaming/session/start
Use	A message sent to the gaming module
References	CS-1, CS-3

Table 16: Properties of Event Topic "GAME_SESSION_START"

Property	Class	Description
INTERACTION_ID	String	A string specifying an interaction id to match responses to requests
SESSION_TYPE	GameSessionType	An enum describing the game session type
SESSION_ID [1]	String	A string describing the game session id
TIMESTAMP	Long	A timestamp representing the number of seconds since 01.01.1970 00:00:00 UTC
USER_ID	String	A string specifying a user id

[1] property is optional

GAME_SESSION_START example.

```
{
  "topic": "eu/vcare/event/gaming/session/start",
  "properties": {
    "INTERACTION_ID": "155742af-4942-4142-8b81-184d2edd1393",
    "TIMESTAMP": 1552320821457,
    "USER_ID": "123456",
    "SESSION_TYPE": "cognitive",
    "SESSION_ID": "567"
  }
}
```

8.1.7 GAME_SESSION_STARTED

This is a message that is sent from the gaming module to indicate a specific session of games was started. Table 17 shows the details and Table 18 shows the properties of this event topic.

Table 17: Details of Event Topic "GAME_SESSION_STARTED"

Name	GAME_SESSION_STARTED
Topic	eu/vcare/event/gaming/session/started
Use	A message sent from the gaming module
References	CS-1, CS-3

Table 18: Properties of Event Topic "GAME_SESSION_STARTED"

Property	Class	Description
INTERACTION_ID	String	A string specifying an interaction id to match responses to requests
SESSION_TYPE	GameSessionType	An enum describing the game session type

SESSION_ID [1]	String	A string describing the game session id
TIMESTAMP	Long	A timestamp representing the number of seconds since 01.01.1970 00:00:00 UTC
USER_ID	String	A string specifying a user id

[1] property is optional

GAME_SESSION_STARTED example.

```
{
  "topic": "eu/vcare/event/gaming/session/started",
  "properties": {
    "INTERACTION_ID": "155742af-4942-4142-8b81-184d2edd1393",
    "TIMESTAMP": 1552320821457,
    "USER_ID": "123456",
    "SESSION_TYPE": "cognitive",
    "SESSION_ID": "567"
  }
}
```

8.1.8 GAME_SESSION_COMPLETED

This is a message that is sent from the gaming module to indicate a specific session of games has been completed. Table 19 shows the details and Table 20 shows the properties of this event topic.

Table 19: Details of Event Topic “GAME_SESSION_COMPLETED”

Name	**GAME_SESSION_COMPLETED**
Topic	eu/vcare/event/gaming/session/completed
Use	A message sent from the gaming module
References	CS-1, CS-3

Table 20: Properties of Event Topic “GAME_SESSION_COMPLETED”

Property	**Class**	**Description**
CONTENT [2]	JsonElement	A JSON object describing the content
INTERACTION_ID	String	A string specifying an interaction id to match responses to requests
SESSION_TYPE	GameSessionType	An enum describing the game session type
SESSION_ID [1]	String	A string describing the game session id

TIMESTAMP	Long	A timestamp representing the number of seconds since 01.01.1970 00:00:00 UTC
USER_ID	String	A string specifying a user id

[1] property is optional
[2] value of the property can be *null*

GAME_SESSION_COMPLETED example.

```
{
  "topic": "eu/vcare/event/gaming/session/completed",
  "properties": {
    "INTERACTION_ID": "155742af-4942-4142-8b81-184d2edd1393",
    "TIMESTAMP": 1552320821447,
    "USER_ID": "123456",
    "SESSION_TYPE": "cognitive",
    "CONTENT": {
      "exit_code": "Can be \"closed\", \"user_aborted\" or \"not_complete\" to
identify how the session has been closed"
    },
    "SESSION_ID": "567"
  }
}
```

8.2 EVENT CATEGORY "HEALTH"

Event topics linked to this category are all related to health services. This category contains the following topics:

Table 21: Event Topics for Event Category "HEALTH"

HEALTH_QUESTIONNAIRE_REQUEST	A message sent to the coaching services
HEALTH_QUESTIONNAIRE_RESULT	A message sent from the coaching services
HEALTH_STATUS_MESSAGE	A message sent to the coaching services
HEALTH_STATUS_REQUEST	A message sent from the coaching services
HEALTH_THERAPY_START	A message sent to the therapy module
HEALTH_THERAPY_STARTED	A message sent from the therapy module

8.2.1 HEALTH_QUESTIONNAIRE_REQUEST

This is a message that is sent to the coaching services to perform a questionnaire for a specific user. Table 22 shows the details and Table 23 shows the properties of this event topic.

Table 22: Details of Event Topic "HEALTH_QUESTIONNAIRE_REQUEST"

Name	**HEALTH_QUESTIONNAIRE_REQUEST**
Topic	eu/vcare/event/health/questionnaire/request
Use	A message sent to the coaching services
References	CS-7

Table 23: Properties of Event Topic "HEALTH_QUESTIONNAIRE_REQUEST"

Property	**Class**	**Description**
CONTENT [2]	JsonElement	A JSON object describing the content
INTERACTION_ID	String	A string specifying an interaction id to match responses to requests
TIMESTAMP	Long	A timestamp representing the number of seconds since 01.01.1970 00:00:00 UTC
USER_ID	String	A string specifying a user id

[2] value of the property can be *null*

HEALTH_QUESTIONNAIRE_REQUEST example.

```
{
  "topic": "eu/vcare/event/health/questionnaire/request",
  "properties": {
    "INTERACTION_ID": "155742af-4942-4142-8b81-184d2edd1393",
    "TIMESTAMP": 1552322003569,
    "USER_ID": "123456",
    "CONTENT": {
      "turns": [
        {
          "id": "Start",
          "content": "This the first question of a demo questionnaire",
          "info": "Some text can also be added as additional information",
          "infoTts": "Some information should only be provided by the text-to-
speech engine.",
          "question": "Why should I answer this questionnaire?",
          "questionTts": "Additional to the question \"Why should I answer this
questionnaire?\" it\u0027s possible to enrich it with TTS pronunciation tags",
          "type": "SINGLE",
          "choices": [
            {
              "content": "Just for fun! Give me all possible questions.",
              "reference": "Question1"
            },
            {
              "content": "I have to do it. I will only answer a few questions...",
              "reference": "Question2"
            },
            {
              "content": "That\u0027s the mandatory third option",
              "reference": "Question3"
            }
          ]
        },
        {
          "id": "Question1",
          "content": "Here you have to pick 3 out of 5...",
          "type": "MULTI",
          "choices": [
            {
              "content": "Answer number one"
            },
            {
              "content": "Answer number two"
            },
            {
              "content": "Answer number three"
            },
            {
              "content": "Answer number four"
            },
            {
              "content": "Answer number five"
            }
          ],
          "minAnswers": 3,
          "reference": "Question2",
          "score": false
        },
        {
          "id": "Question2",
          "question": "Please pick one answer",
          "type": "SINGLE",
          "choices": [
            {
              "content": "Answer number one"
            },
            {
              "content": "Answer number two"
            },
            {
              "content": "Answer number three"
            },
            {
              "content": "Answer number four"
            }
          ],
          "reference": "Question3",
          "score": true
        },
        {
          "id": "Question3",
          "question": "On a scale of 1 to 10, how do you feel today? (1 \u003d
very bad, 5 \u003d as usual, 10 \u003d very good)",
          "type": "INT_SLIDER",
          "score": true,
          "slider": {
            "min": 1,
            "max": 10
          },
          "reference": "Question4"
        },
        {
          "id": "Question4",
          "precondition": "@{Question3} \u003c 5\n",
          "question": "Why do you feel bad today? Remember that you selected
@{Question3} in the previous question. Why didn\u0027t you select @[@{Question3} +
3]",
          "type": "STRING",
          "reference": "Question5"
        },
        {
          "id": "Question5",
          "precondition": "@{Question3} \u003e 5\n",
          "question": "Why do you feel good today? Remember that you selected
@{Question3} in the previous question. Why didn\u0027t you select @[@{Question3} -
4]",
          "type": "STRING",
          "reference": "End"
        },
        {
          "id": "End",
          "question": "We are done!",
          "type": "INFO",
          "score": false
        }
      ]
    }
  }
}
```

8.2.2 HEALTH_QUESTIONNAIRE_RESULT

This is a message that is sent from the coaching services with the result of a questionnaire for a specific user. Table 24 shows the details and Table 25 shows the properties of this event topic.

Table 24: Details of Event Topic "HEALTH_QUESTIONNAIRE_RESULT"

Name	**HEALTH_QUESTIONNAIRE_RESULT**
Topic	eu/vcare/event/health/questionnaire/result
Use	A message sent from the coaching services
References	CS-7

Table 25: Properties of Event Topic "HEALTH_QUESTIONNAIRE_RESULT"

Property	**Class**	**Description**
CONTENT [2]	JsonElement	A JSON object describing the content
INTERACTION_ID	String	A string specifying an interaction id to match responses to requests
TIMESTAMP	Long	A timestamp representing the number of seconds since 01.01.1970 00:00:00 UTC
USER_ID	String	A string specifying a user id

[2] value of the property can be *null*

HEALTH_QUESTIONNAIRE_RESULT example.

```
{
  "topic": "eu/vcare/event/health/questionnaire/result",
  "properties": {
    "INTERACTION_ID": "155742af-4942-4142-8b81-184d2edd1393",
    "TIMESTAMP": 1552322003570,
    "USER_ID": "123456",
    "CONTENT": {
      "turns": [
        {
          "id": "Start",
          "type": "SINGLE",
          "answer": 2
        },
        {
          "id": "Question1",
          "type": "MULTI",
          "answers": "1, 3, 5"
        },
        {
          "id": "Question2",
          "type": "SINGLE",
          "answer": 4
        },
        {
          "id": "Question3",
          "type": "INT_SLIDER",
          "answer": 2
        },
        {
          "id": "Question4",
          "type": "STRING",
          "answer": "I\u0027ve no clue."
        }
      ]
    }
  }
}
```

8.2.3 HEALTH_STATUS_MESSAGE

This is a message that is sent to the coaching services with the health status for a specific user.

Table 26 shows the details and Table 27 shows the properties of this event topic.

Table 26: Details of Event Topic "HEALTH_STATUS_MESSAGE"

Name	HEALTH_STATUS_MESSAGE
Topic	eu/vcare/event/health/status/message
Use	A message sent to the coaching services
References	CS-2, DS-5, DS-6, DS-7, DS-9

Table 27: Properties of Event Topic "HEALTH_STATUS_MESSAGE"

Property	Class	Description
CONTENT [2]	JsonElement	A JSON object describing the content
INTERACTION_ID	String	A string specifying an interaction id to match responses to requests
TIMESTAMP	Long	A timestamp representing the number of seconds since 01.01.1970 00:00:00 UTC
USER_ID	String	A string specifying a user id

[2] value of the property can be *null*

HEALTH_STATUS_MESSAGE example.

```
{
  "topic": "eu/vcare/event/health/status/message",
  "properties": {
    "INTERACTION_ID": "155742af-4942-4142-8b81-184d2edd1393",
    "TIMESTAMP": 1552322003570,
    "USER_ID": "123456",
    "CONTENT": [
      {
        "Type": "BMI",
        "Value": "22",
        "Unit": "kg/m^2",
        "Reward": 1
      }
    ]
  }
}
```

8.2.4 HEALTH_STATUS_REQUEST

This is a message that is sent from the coaching services to request the health status for a specific user. Table 28 shows the details and Table 29 shows the properties of this event topic.

Table 28: Details of Event Topic "HEALTH_STATUS_REQUEST"

Name	**HEALTH_STATUS_REQUEST**
Topic	eu/vcare/event/health/status/request
Use	A message sent from the coaching services
References	CS-2, DS-5, DS-6, DS-7, DS-9

Table 29: Properties of Event Topic "HEALTH_STATUS_REQUEST"

Property	**Class**	**Description**
CONTENT [2]	JsonElement	A JSON object describing the content
INTERACTION_ID	String	A string specifying an interaction id to match responses to requests
TIMESTAMP	Long	A timestamp representing the number of seconds since 01.01.1970 00:00:00 UTC
USER_ID	String	A string specifying a user id

[2] value of the property can be *null*

HEALTH_STATUS_REQUEST example.

```
{
  "topic": "eu/vcare/event/health/status/request",
  "properties": {
    "INTERACTION_ID": "155742af-4942-4142-8b81-184d2edd1393",
    "TIMESTAMP": 1552322003571,
    "USER_ID": "123456",
    "CONTENT": {
      "Types": [
        "BMI"
      ]
    }
  }
}
```

8.2.5 HEALTH_THERAPY_START

This is a message that is sent to the therapy module to start a specific therapy. Table 30 shows the details and Table 31 shows the properties of this event topic.

Table 30: Details of Event Topic "HEALTH_THERAPY_START"

Name	HEALTH_THERAPY_START
Topic	eu/vcare/event/health/therapy/start
Use	A message sent to the therapy module
References	CS-9

Table 31: Properties of Event Topic "HEALTH_THERAPY_START"

Property	Class	Description
INTERACTION_ID	String	A string specifying an interaction id to match responses to requests
THERAPY_TYPE	TherapyType	An enum describing the therapy type
THERAPY_ID [1]	String	A string describing the game id
TIMESTAMP	Long	A timestamp representing the number of seconds since 01.01.1970 00:00:00 UTC
USER_ID	String	A string specifying a user id

[1] property is optional

HEALTH_THERAPY_START example.

```
{
  "topic": "eu/vcare/event/health/therapy/start",
  "properties": {
    "INTERACTION_ID": "155742af-4942-4142-8b81-184d2edd1393",
    "THERAPY_TYPE": "speech-swallowing",
    "THERAPY_ID": "890",
    "TIMESTAMP": 1552322003571,
    "USER_ID": "123456"
  }
}
```

8.2.6 HEALTH_THERAPY_STARTED

This is a message that is sent from the therapy module to indicate a specific therapy was started. Table 32 shows the details and Table 33 shows the properties of this event topic.

Table 32: Details of Event Topic "HEALTH_THERAPY_STARTED"

Name	**HEALTH_THERAPY_STARTED**
Topic	eu/vcare/event/health/therapy/started
Use	A message sent from the therapy module
References	CS-9

Table 33: Properties of Event Topic "HEALTH_THERAPY_STARTED"

Property	**Class**	**Description**
INTERACTION_ID	String	A string specifying an interaction id to match responses to requests
THERAPY_TYPE	TherapyType	An enum describing the therapy type
THERAPY_ID [1]	String	A string describing the game id
TIMESTAMP	Long	A timestamp representing the number of seconds since 01.01.1970 00:00:00 UTC
USER_ID	String	A string specifying a user id

[1] property is optional

HEALTH_THERAPY_STARTED example.

```
{
  "topic": "eu/vcare/event/health/therapy/started",
  "properties": {
    "INTERACTION_ID": "155742af-4942-4142-8b81-184d2edd1393",
    "THERAPY_TYPE": "speech-swallowing",
    "THERAPY_ID": "890",
    "TIMESTAMP": 1552322003571,
    "USER_ID": "123456"
  }
}
```

8.3 EVENT CATEGORY "LOCATION"

Event topics linked to this category are all related to the location of a user. This category contains the following topics:

Table 34: Event Topics for Event Category "LOCATION"

LOCATION_MESSAGE	A message sent from the data services
LOCATION_REQUEST	A message sent to the data services

8.3.1 LOCATION_MESSAGE

This is a message that is sent from the data services with the location for a specific user. Table 35 shows the details and Table 36 shows the properties of this event topic.

Table 35: Details of Event Topic “LOCATION_MESSAGE”

Name	**LOCATION_MESSAGE**
Topic	eu/vcare/event/location/message
Use	A message sent from the data services
References	DS-1, DS-8

Table 36: Properties of Event Topic “LOCATION_MESSAGE”

Property	**Class**	**Description**
CONTENT [2]	JsonElement	A JSON object describing the content
INTERACTION_ID	String	A string specifying an interaction id to match responses to requests
TIMESTAMP	Long	A timestamp representing the number of seconds since 01.01.1970 00:00:00 UTC
USER_ID	String	A string specifying a user id

[2] value of the property can be *null*

LOCATION_MESSAGE example.

```
{
  "topic": "eu/vcare/event/location/message",
  "properties": {
    "INTERACTION_ID": "155742af-4942-4142-8b81-184d2edd1393",
    "TIMESTAMP": 1553074687213,
    "USER_ID": "123456",
    "CONTENT": [
      {
        "Type": "GPS",
        "Lat": "N48Â°16\u002703.60\"",
        "Lon": "E16Â°22\u002717.00\""
      }
    ]
  }
}
```

8.3.2 LOCATION_REQUEST

This is a message that is sent to the data services to request the location for a specific user. Table 37 shows the details and Table 38 shows the properties of this event topic.

Table 37: Details of Event Topic "LOCATION_REQUEST"

Name	LOCATION_REQUEST
Topic	eu/vcare/event/location/request
Use	A message sent to the data services
References	DS-1, DS-8

Table 38: Properties of Event Topic "LOCATION_REQUEST"

Property	Class	Description
CONTENT [2]	JsonElement	A JSON object describing the content
INTERACTION_ID	String	A string specifying an interaction id to match responses to requests
TIMESTAMP	Long	A timestamp representing the number of seconds since 01.01.1970 00:00:00 UTC
USER_ID	String	A string specifying a user id

[2] value of the property can be `null`

LOCATION_REQUEST example.

```
{
  "topic": "eu/vcare/event/location/request",
  "properties": {
    "INTERACTION_ID": "155742af-4942-4142-8b81-184d2edd1393",
    "TIMESTAMP": 1553074687214,
    "USER_ID": "123456",
    "CONTENT": {
      "Types": [
        "GPS"
      ]
    }
  }
}
```

8.4 EVENT CATEGORY "MESSAGE"

Event topics linked to this category are all related to general messages. This category contains the following topics:

Table 39: Event Topics for Event Category "MESSAGE"

USER_MESSAGE	A message sent to the coaching services

8.4.1 USER_MESSAGE

This is a message that is sent to the coaching services to inform the user. Table 40 shows the details and Table 41 shows the properties of this event topic.

Table 40: Details of Event Topic "USER_MESSAGE"

Name	**USER_MESSAGE**
Topic	eu/vcare/event/message/user
Use	A message sent to the coaching services
References	CS-4

Table 41: Properties of Event Topic "USER_MESSAGE"

Property	**Class**	**Description**
CONTENT [2]	JsonElement	A JSON object describing the content
INTERACTION_ID	String	A string specifying an interaction id to match responses to requests
TIMESTAMP	Long	A timestamp representing the number of seconds since 01.01.1970 00:00:00 UTC
USER_ID	String	A string specifying a user id

[2] value of the property can be *null*

USER_MESSAGE example.

```
{
  "topic": "eu/vcare/event/message/user",
  "properties": {
    "INTERACTION_ID": "155742af-4942-4142-8b81-184d2edd1393",
    "TIMESTAMP": 1552322003576,
    "USER_ID": "123456",
    "CONTENT": {
      "content": "not defined yet"
    }
  }
}
```

8.5 Event Category "POSITION"

Event topics linked to this category are all related to the position of a user. This category contains the following topics:

Table 42: Event Topics for Event Category "POSITION"

POSITION_MESSAGE	A message sent from the data services
POSITION_REQUEST	A message sent to the data services

8.5.1 POSITION_MESSAGE

This is a message that is sent from the data services with the position for a specific user. Table 43 shows the details and Table 44 shows the properties of this event topic.

Table 43: Details of Event Topic "POSITION_MESSAGE"

Name	POSITION_MESSAGE
Topic	eu/vcare/event/position/message
Use	A message sent from the data services
References	DS-2, DS-4

Table 44: Properties of Event Topic "POSITION_MESSAGE"

Property	Class	Description
CONTENT [2]	JsonElement	A JSON object describing the content
INTERACTION_ID	String	A string specifying an interaction id to match responses to requests
TIMESTAMP	Long	A timestamp representing the number of seconds since 01.01.1970 00:00:00 UTC
USER_ID	String	A string specifying a user id

[2] value of the property can be *null*

POSITION_MESSAGE example.

```
{
  "topic": "eu/vcare/event/position/message",
  "properties": {
    "INTERACTION_ID": "155742af-4942-4142-8b81-184d2edd1393",
    "TIMESTAMP": 1553074687219,
    "USER_ID": "123456",
    "CONTENT": [
      {
        "Type": "Body",
        "Position": "Standing"
      }
    ]
  }
}
```

8.5.2 POSITION_REQUEST

This is a message that is sent to the data services to request the position for a specific user. Table 45 shows the details and Table 46 shows the properties of this event topic.

Table 45: Details of Event Topic "POSITION_REQUEST"

Name	POSITION_REQUEST
Topic	eu/vcare/event/position/request
Use	A message sent to the data services
References	DS-2, DS-4

Table 46: Properties of Event Topic "POSITION_REQUEST"

Property	Class	Description
CONTENT [2]	JsonElement	A JSON object describing the content
INTERACTION_ID	String	A string specifying an interaction id to match responses to requests
TIMESTAMP	Long	A timestamp representing the number of seconds since 01.01.1970 00:00:00 UTC
USER_ID	String	A string specifying a user id

[2] value of the property can be *null*

POSITION_REQUEST example.

```
{
  "topic": "eu/vcare/event/position/request",
  "properties": {
    "INTERACTION_ID": "155742af-4942-4142-8b81-184d2edd1393",
    "TIMESTAMP": 1553074687220,
    "USER_ID": "123456",
    "CONTENT": {
      "Types": [
        "Body"
      ]
    }
  }
}
```

8.6 Event Category "REMINDER"

Event topics linked to this category are all related to reminders. This category contains the following topics:

Table 47: Event Topics for Event Category "REMINDER"

REMINDER_MESSAGE	A message sent to the coaching services
REMINDER_POSTPONE	A message sent from the coaching services

8.6.1 REMINDER_MESSAGE

This is a message that is sent to the coaching services to remind the user e.g. about a specific task. Table 48 shows the details and Table 49 shows the properties of this event topic.

Table 48: Details of Event Topic "REMINDER_MESSAGE"

Name	REMINDER_MESSAGE
Topic	eu/vcare/event/reminder/message
Use	A message sent to the coaching services
References	CS-1, CS-3, CS-6, CS-8, CS-9

Table 49: Properties of Event Topic "REMINDER_MESSAGE"

Property	Class	Description
CONTENT [2]	JsonElement	A JSON object describing the content
INTERACTION_ID	String	A string specifying an interaction id to match responses to requests
REMINDER_TYPE	ReminderType	An enum describing the reminder type
TIMESTAMP	Long	A timestamp representing the number of seconds since 01.01.1970 00:00:00 UTC
USER_ID	String	A string specifying a user id

[2] value of the property can be `null`

REMINDER_MESSAGE example.

```
{
  "topic": "eu/vcare/event/reminder/message",
  "properties": {
    "INTERACTION_ID": "155742af-4942-4142-8b81-184d2edd1393",
    "TIMESTAMP": 1552322003581,
    "USER_ID": "123456",
    "CONTENT": {
      "content": "not defined yet"
    },
    "REMINDER_TYPE": "medication"
  }
}
```

8.6.2 REMINDER_POSTPONE

This is a message that is sent from the coaching services to indicate that a reminder should be postponed. Table 50 shows the details and Table 51 shows the properties of this event topic.

Table 50: Details of Event Topic "REMINDER_POSTPONE"

Name	REMINDER_POSTPONE
Topic	eu/vcare/event/reminder/postpone
Use	A message sent from the coaching services
References	CS-1, CS-3, CS-6, CS-9

Table 51: Properties of Event Topic "REMINDER_POSTPONE"

Property	Class	Description
CONTENT [2]	JsonElement	A JSON object describing the content
INTERACTION_ID	String	A string specifying an interaction id to match responses to requests
REMINDER_TYPE	ReminderType	An enum describing the reminder type
TIMESTAMP	Long	A timestamp representing the number of seconds since 01.01.1970 00:00:00 UTC
USER_ID	String	A string specifying a user id

[2] value of the property can be *null*

REMINDER_POSTPONE example.

```
{
  "topic": "eu/vcare/event/reminder/postpone",
  "properties": {
    "INTERACTION_ID": "155742af-4942-4142-8b81-184d2edd1393",
    "TIMESTAMP": 1552322003581,
    "USER_ID": "123456",
    "REMINDER_TYPE": "medication"
  }
}
```

8.7 Event Category "SUPPORT"

Event topics linked to this category are all related to the supporting services. This category contains the following topics:

Table 52: Event Topics for Event Category "SUPPORT"

SUPPORT_SERVICE_MESSAGE	A message sent from the supporting services
SUPPORT_SERVICE_REQUEST	A message sent to the supporting services

8.7.1 SUPPORT_SERVICE_MESSAGE

This is a message that is sent from the supporting services with data for a specific user. Table 53 shows the details and Table 54 shows the properties of this event topic.

Table 53: Details of Event Topic “SUPPORT_SERVICE_MESSAGE”

Name	SUPPORT_SERVICE_MESSAGE
Topic	eu/vcare/event/support/service/message
Use	A message sent from the supporting services
References	SS-1, SS-2, SS-3

Table 54: Properties of Event Topic “SUPPORT_SERVICE_MESSAGE”

Property	Class	Description
CONTENT [2]	JsonElement	A JSON object describing the content
INTERACTION_ID	String	A string specifying an interaction id to match responses to requests
TIMESTAMP	Long	A timestamp representing the number of seconds since 01.01.1970 00:00:00 UTC
USER_ID	String	A string specifying a user id

[2] value of the property can be `null`

SUPPORT_SERVICE_MESSAGE example.

```
{
  "topic": "eu/vcare/event/support/service/message",
  "properties": {
    "INTERACTION_ID": "155742af-4942-4142-8b81-184d2edd1393",
    "TIMESTAMP": 1553075334185,
    "USER_ID": "123456",
    "CONTENT": [
      {
        "Type": "WeatherForecastToday",
        "Data": {
          "coord": {
            "lon": 139,
            "lat": 35
          },
          "sys": {
            "country": "JP",
            "sunrise": 1369769524,
            "sunset": 1369821049
          },
          "weather": [
            {
              "id": 804,
              "main": "clouds",
              "description": "overcast clouds",
              "icon": "04n"
            }
          ],
          "main": {
            "temp": 289.5,
            "humidity": 89,
            "pressure": 1013,
            "temp_min": 287.04,
            "temp_max": 292.04
          },
          "wind": {
            "speed": 7.31,
            "deg": 187.002
          },
          "rain": {
            "3h": 0
          },
          "clouds": {
            "all": 92
          },
          "dt": 1369824698,
          "id": 1851632,
          "name": "Vienna",
          "cod": 200
        }
      }
    ]
  }
}
```

8.7.2 SUPPORT_SERVICE_REQUEST

This is a message that is sent to the supporting services to request data for a specific user. Table 55 shows the details and Table 56 shows the properties of this event topic.

Table 55: Details of Event Topic “SUPPORT_SERVICE_REQUEST”

Name	**SUPPORT_SERVICE_REQUEST**
Topic	eu/vcare/event/support/service/request
Use	A message sent to the supporting services
References	SS-1, SS-2, SS-3

Table 56: Properties of Event Topic “SUPPORT_SERVICE_REQUEST”

Property	**Class**	**Description**
CONTENT [2]	JsonElement	A JSON object describing the content
INTERACTION_ID	String	A string specifying an interaction id to match responses to requests
TIMESTAMP	Long	A timestamp representing the number of seconds since 01.01.1970 00:00:00 UTC
USER_ID	String	A string specifying a user id

[2] value of the property can be `null`

SUPPORT_SERVICE_REQUEST example.

```
{
  "topic": "eu/vcare/event/support/service/request",
  "properties": {
    "INTERACTION_ID": "155742af-4942-4142-8b81-184d2edd1393",
    "TIMESTAMP": 1553075334186,
    "USER_ID": "123456",
    "CONTENT": {
      "Types": [
        "WeatherForecastToday"
      ]
    }
  }
}
```

9 IDENTIFICATION AND AUTHENTICATION

Within vCare identification and authentication is realized by using Keycloak in combination with the KIOLA platform. KIOLA is part of the security components layer of the overall vCare system architecture, which uses Keycloak as a single-sign-on solution for all services invoked in vCare. Furthermore, KIOLA provides an identity management component and a role and user management system for all services ensuring that only authorized stakeholders can access data of patients.

For security purposes tokens, provided by Keycloak, following the JWT specification[8] are used for authentication and authorization between the various vCare services and components.

As data is stored in different databases at various layers of the architecture (e.g. the middleware, the knowledge and coaching layer) vCare requires an identity management component, which is provided by the KIOLA platform. A RESTful interface is provided for the various platforms of the vCare platform that enables clients to register local data identifiers through a HTTP POST request. KIOLA is then able to link all known identifiers through the Keycloak user account. Moreover, KIOLA provides an interface to map and retrieve identifiers through a HTTP GET request as required.

Permissions in KIOLA are organized around so-called user circles, illustrated in Figure 8. User circles define a number of actors (e.g., general practitioners, clinical experts, coordinators) and an audience (patients). Actors within a user circle are allowed to view and modify data of their audience (patients). Typical examples of user circles include: hospitals, medical office or even entire regions (e.g., a country, a city or a federal state):

[8] https://tools.ietf.org/html/rfc7519

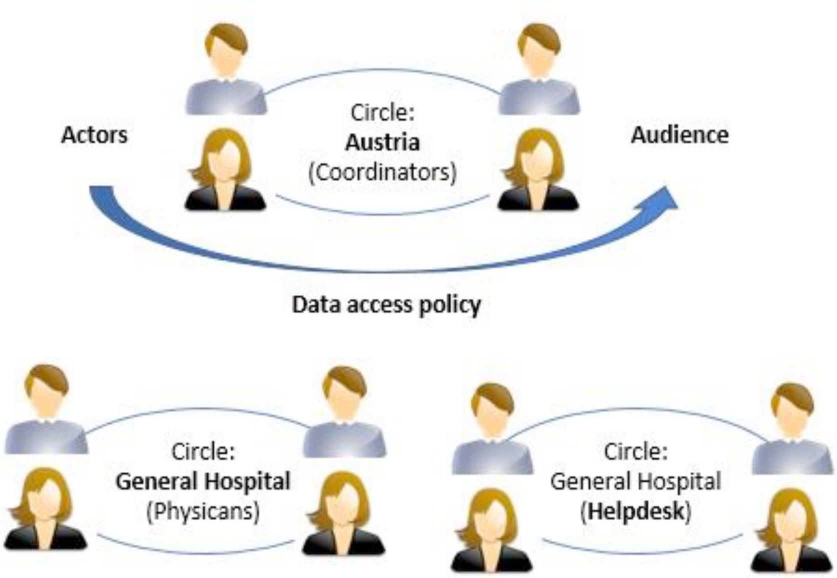


Figure 8: Circle permission system: Circles are defined by actors (users who can access data), audience (typically patients whose data is accessed), the role (e.g. coordinator) which defines the data access policy (what data can be defined).

User registration can be performed only by coordinators and includes three steps:

- providing personal data
- patient assignment to a circle (thus, to a clinic/medical office)
- review of the data provided

10 MULTI-LANGUAGE SUPPORT

Within the vCare project, support of the languages Danish, Romanian, Spanish and Italian is required. In addition, English will be supported as base language as well. Even there are no trials in English, it simplifies the translation to the other languages and will be also used as input language for certain services, e.g. for the IBM Watson natural language understanding API.

For translating the content of the services, the “GNU gettext” API will be used. In computing, gettext is an internationalization (i18n) and localization (i10n) system commonly used for writing multilingual programs on Unix-like computer operating systems. The most commonly used implementation of gettext is “GNU gettext” released by the GNU Project in 1995[9].

Gettext is also well supported in the Java programming language. Together with related Maven plugins[10], standardized portable object (PO) and Portable Object Template (POT) files are generated automatically from the source code, using the English version of the text to translate as key. Dynamic content is provided as well, e.g. the code

```
i18n.tr("The weather in {0} today is {1}. The temperature ranges from {2} to {3} degrees.",
objs);
```

returns the weather information with variables in the requested language as provided by the localization settings of the installation.

For each language a separate PO file will be created and translated using an appropriate editor, e.g. Poedit[11] (see Figure 9).

Poedit provides translators and developers with a powerful and intuitive editor for gettext. It helps to save time on mundane translation tasks with a lightweight and easy-to-use interface and smart features like pre-translation and machine translation.

Summarized, the translation procedure in short works as follows:

1. The developer writes their code using gettext APIs.
2. The developer extracts all translatable texts in their code and creates a master PO file that translators can work with.
3. The translator works through the PO file provided and hands back their translations, one file per language.
4. Whenever the source code changes and there are translation differences the developer re-runs the extraction and updates the master PO file accordingly.
5. The translator only has to update their translation file to match the master file 100%.
6. Poedit can help with stages 2 through 5 and save both, translators and developers, time at each stage, thanks to its focused workflow and extensive capabilities.

[9] https://www.gnu.org/software/gettext/manual/gettext.html#Introduction-1-1e

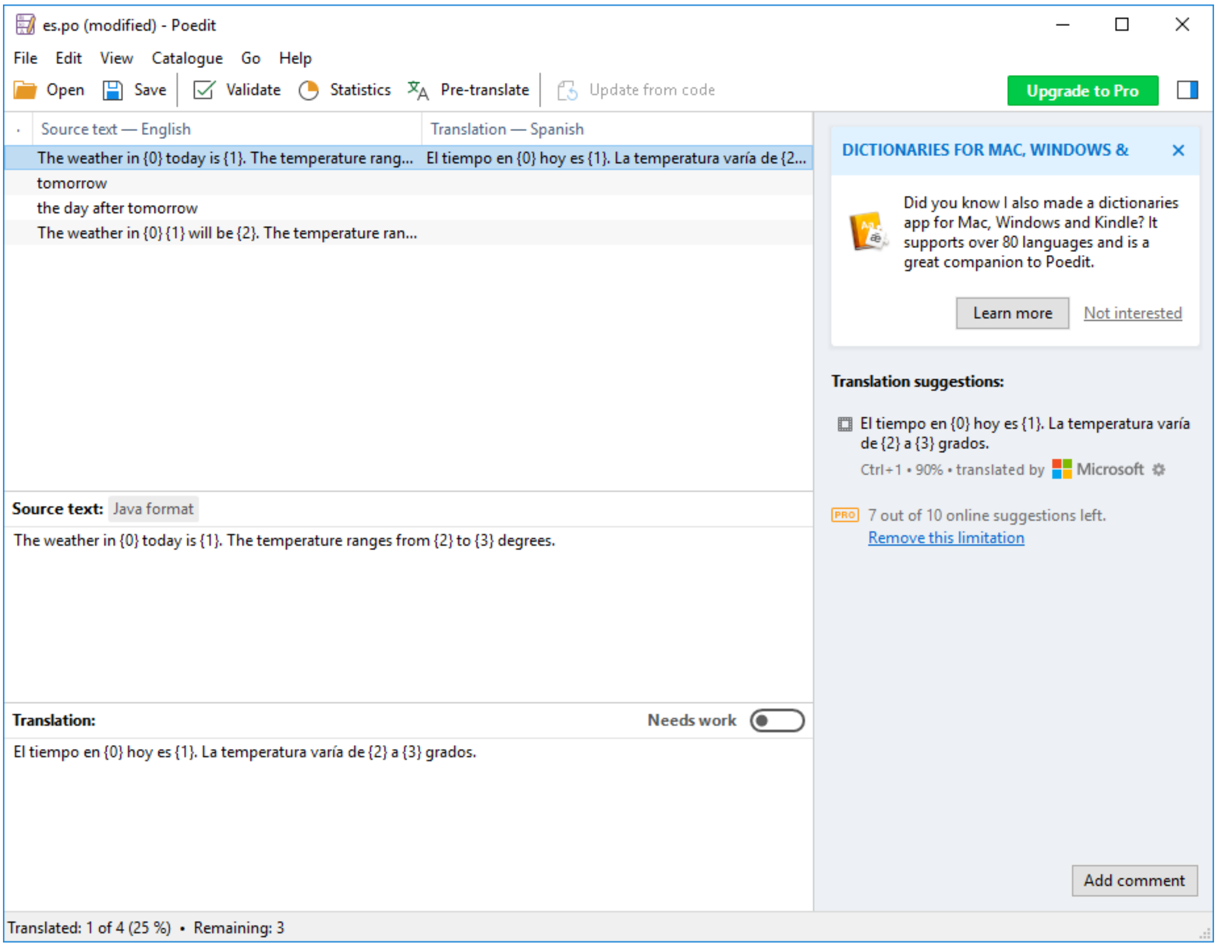
[10] e.g. https://github.com/jgettext/gettext-commons

[11] https://poedit.net/

Within vCare, translators for each language are defined and listed in Table 57.

Table 57: Language translators

Language	Translator	Partner
Spanish	Jordi Rovira	MYS
Italian	Rosangela Boninsegna	CCP
Romanian	Mircea Vasile	SIV
Danish	Sadia Anwar	AU



es.po (modified) - Poedit

Source text — English	Translation — Spanish
The weather in {0} today is {1}. The temperature rang...	El tiempo en {0} hoy es {1}. La temperatura varía de {2...
tomorrow	
the day after tomorrow	
The weather in {0} {1} will be {2}. The temperature ran...	

DICTIONARIES FOR MAC, WINDOWS &

Did you know I also made a dictionaries app for Mac, Windows and Kindle? It supports over 80 languages and is a great companion to Poedit.

Translation suggestions:

El tiempo en {0} hoy es {1}. La temperatura varía de {2} a {3} grados.

Ctrl+1 • 90% • translated by Microsoft

PRO 7 out of 10 online suggestions left.

Remove this limitation

Source text: Java format

The weather in {0} today is {1}. The temperature ranges from {2} to {3} degrees.

Translation:

El tiempo en {0} hoy es {1}. La temperatura varía de {2} a {3} grados.

Translated: 1 of 4 (25 %) • Remaining: 3

Figure 9: Screenshot of the Poedit tool

11 Conclusion and Outlook

This deliverable provides a comprehensive overview of the services and their interfaces which will be provided by vCare and reflect the requirements as defined in WP1 and WP7. In addition to a functional description, the required personal data as well as the integration in the overall architecture is provided. The service specific architecture shows a seamless integration within the vCare system by leveraging the MQTT protocol and its publish-subscribe based messaging mechanism. This provides a high level of flexibility and extensibility of vCare's core system, which makes it easy to enhance its functionalities and features by connecting new components or services to the MQTT broker. Likewise, the concept of the vCare-as-a-Service paradigm enables third parties to provide additional features to their platform, effectively extending their service portfolio by using the vCare-as-a-Service REST API. This concept is useful for exploiting vCare as it shows its openness to other applications.

Most importantly, this document provides a detailed specification and description of the message (event) format along with event topics and event categories, which is a reference for all vCare components how to communicate with each other through the MQTT broker. This specification includes a clear description on what events, event topics and event categories are, what they are used for and how they look like. Additionally, examples are given on how JSON encoded events will look like. Therefore, this document provides essential information for the further software development and integration phase.

Finally, an overview about the authentication and authorization mechanism within the vCare system is given, by using the well-established JWT standard. For managing the multi-language aspect of vCare, the project makes use of the commonly used gettext system together with a translation tool like Poedit.

This deliverable will be followed up by D5.3 "Coaching services description", which will provide a thorough description and in-depth implementation details of the services defined in D5.1 together with their interfaces specified in the current document. Likewise to this deliverable, D5.3 will address not only the coaching services as stated in its title, but also the data and supporting services.