WP3 – SMART ADAPTIVE CASE MANAGEMENT

D3.1: SOCIOCORTEX FOR HEALTHCARE

H2020-EU.3.1: Personalised Connected Care for Complex Chronic Patients

Project No. 689802

Start date of project: 01-04-2016

Duration: 42 months

Project funded by the European Commission, call H2020 – PHC - 2015	
✓ PU	Public
PP	Restricted to other programme participants (including the Commission Services)
RE	Restricted to a group specified by the consortium (including the Commission Services)
CO	Confidential, only for members of the consortium (including the Commission Services)

Revision: 01

Date: 30-9-2017

Document Information

Project Number	689802	**Acronym**	CONNECARE
Full title	Personalised Connected Care for Complex Chronic Patients		
Project URL	http://www.CONNECARE.eu		

Deliverable	**Number**	D3.1	**Title**	SocioCortex for healthcare (formely: Darwin for healthcare)
Work Package	**Number**	3	**Title**	Smart Adaptive Case Management

Date of delivery	**Contractual**		**Actual**	
Nature	Prototype ☑ Report ☐ Dissemination ☐ Other ☐			
Dissemination Level	Public ☐ Consortium ☑			

Responsible Author	Felix Michel	**Email**	felix.michel@tum.de
Partner	TUM	**Phone**	+49 89 289-17129

Abstract	This document describes the technical changes that are applied on the SocioCortex Platform to support the requirements from CONNECARE. The resulting system is called the SACM-Backend and uses an enhanced version of SocioCortex as core. SocioCortex has been iteratively extended according to support healthcare specific requirements. The following sections describe: 1) the architecture including the integration scenarios, 2) SACM meta-model elements, 3) the case modelling procedure and 4) the software documentation.

CONNECARE project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement Nº 689802 (Art. 29.4)

Table of contents

Executive Summary

A system for Smart Adaptive Case Management (SACM) is an integral part of CONNECARE and is described in D3.3: FIRST SMART ADAPTIVE CASE MANAGEMENT SYSTEM. The core of the SACM system is provided by SocioCortex which is actively developed at TUM. Historically, the predecessor of SocioCortex was called Darwin. However, Darwin did not fulfil all requirements that are necessary to be used as a production system in a healthcare domain like CONNECARE. SocioCortex incorporates extended functionalities for case modelling and execution, specifically tailored to the healthcare domain. A comprehensive description of the organisational and technological dimensions of ACM in healthcare is described in D2.2 ACM DESIGN. Relevant requirements are described in D2.4 CASE STUDY DESCRIPTION. Additionally, SocioCortex implements advanced access control methods, a higher-order functional language to compute derived attributes on the fly and more sophisticated search and indexing features.

The SACM system developed for CONNECARE integrates, in addition to the SocioCortex core, a decision support system (DSS) for clinicians and a professional user interface that makes it easy for them to interact with the system.

The SACM is only one part of the whole CONNECARE system. There is also a separate system for user identity management and one for self-management of the patient (SMS). All three systems communicate via a REST API over a message broker. SocioCortex is designed to communicate with multiple different systems at once and therefore provides a special REST API which allows third-party applications like the SMS to retrieve and send data like case data and patient data. The models of the cases (CaseDefinitions) and the data (DataDefininitions) are defined in cooperation with the domain experts, i.e. clinicians, and are then imported into SocioCortex.

This detailed technical document describes the SACM architecture including the integration scenarios, the SACM meta-model elements, the case modelling procedure and gives an overview of the online software documentation.

Overall, the work summarized in this document is based on the work made by TUM in WP3 with the support of EURECAT. According to the CONNECARE fundamentals, that work has been done in collaboration with clinical partners and decisions came from the co-design approach followed in WP2. Therefore, these previous deliverables are highly recommended to be read:

Number	Title	Description
D2.1	Cook-book for project development	The document provides an overall view of the CONNECARE project, and describes the procedures for its development. The deliverable indicates the

		different phases of the project, with an emphasis on how PDSA cycles will be structured. Overall, the CONNECARE project does not aim at a rigid integrated care solution that needs to be adopted by all potential deployment sites but to a flexible solution that has high potential for generalization at the EU level. In this sense, innovative methodologies involving both global and local stakeholders have been adopted.
D2.2	Adaptive Case Management Design	The main aim of this document is to provide a comprehensive description of the organisational and technological dimensions of ACM in healthcare that have been taken into account in the co-design of the functional requirements of the CONNECARE smart ACM platform. In addition, existing site-specific organisational and technological settings within which CONNECARE case studies and technologies will be implemented, are described. Ultimately, this document will be input to the co-design process and the functional specifications of CONNECARE integrated care services and supporting adaptive case management technologies.
D2.4	Case studies description and the associated co-design	The document provides a complete view of case study definitions as a product of the co-design process completed so far. It provides full details on the 1st PDSA cycle from the clinical perspective, summarizing the objectives and results of all held meetings and activities, as well as all the feedback provided to technical partners. Moreover, the current document includes detailed site-specific case studies definitions and associated workflows. Finally, full details on functional and non-functional requirements of the CONNECARE Smart Adaptive Case Management (SACM) platform And Self-Management System (SMS) are provided.
D5.1	Collaborative Digital Health Framework	This deliverable describes the collaborative DHF that includes the interoperability model and the communication protocols.

Moreover, also these deliverables ready at M18 (September 2017) as the current one, are recommended to be read:

Number	**Title**	**Description**
D3.2	First Screening and Risk Stratification DSS	This deliverable has the twofold goal of (i) reporting on the development activities carried out to deliver the 1st prototype of the risk assessment DSS for screening and risk stratification, as well as of (ii) describing the resulting software artefact. Accordingly, section 1 motivates and gives context to the work done, section 2 summarises the requirement collection phase, section 3 describes the DSS architecture, section 4 provides technical details on the implementation, section 5 looks forward to future iterative improvement steps, and section 6 concludes the document.

D3.3	D3.3_First Smart Adaptive Case Management System_v4	This deliverable goes with the first release (namely, Study Release) of the Smart Adaptive Case Management system (SACM) by TUM and ADI, under the supervision of EURECAT and the contribution of UNIMORE for the first clinical decision support system.
D5.2	Study Release of the generic CONNECARE system	In this document, the first release (Study Release) of the generic CONNECARE system is presented. Its customization to each site of the project is documented in deliverables D5.4, D5.6, and D5.8.

1. Architecture

The Smart Adaptive Case Management (SACM) is composed of several subcomponents and is connected by the CONNECARE Message Broker (see Figure 1). In general, SocioCortex[1] provides the basic Adaptive Case Management (ACM) functionality enriched with the Decisions Support System (DSS) to provide SACM functionality. Clinicians can access the SACM system via a Professional User Interface. This document focuses primary on the extension of SocioCortex for healthcare, anyway all components are explained at high-level in order to provide an overview:

SocioCortex-Server has several conceptual layers that are accessible by a JSON based REST API. Beginning from the bottom, the layers are: 1) Versioned Linked Content Graph, all content is stored in a linked graph and with version control, 2) Dynamic Schemata, models can be adjusted during runtime, 3) Role Based Access Control, on artefacts based on groups, 4) Advanced Search and Indexing, allows full text search on artefacts, 5) Higher-Order Functional Language, to write complex queries on the content graph, 6) Case Modelling, provides the ACM functionality, 7) Case Execution, allows to instantiate and execute the modelled cases. The case data is stored in a local MySQL database. (Java 1.6, MySQL)

SACM-Backend provides a domain specific functionality as well as an API for the Professional Interface and the CONNNECARE Message Broker. Internally, the ACM functionality of the SocioCortex-Server is wrapped and enhanced to provide the domain specific functionality. (NodeJS 4.x)

Decision Support System support information for clinicians. (Java)

Professional Interface provides information needed by professionals such as doctors, nurses etc. For simplification, the first prototype of this interface only executes workflows that are modelled based on a XML file. The client of this interface uses the domain-specific API. (Angular 2.0)

[1]The SocioCortex Software is the successor of the Darwin Software both developed by TUM. It incorporates the changes for healthcare as described in this document.

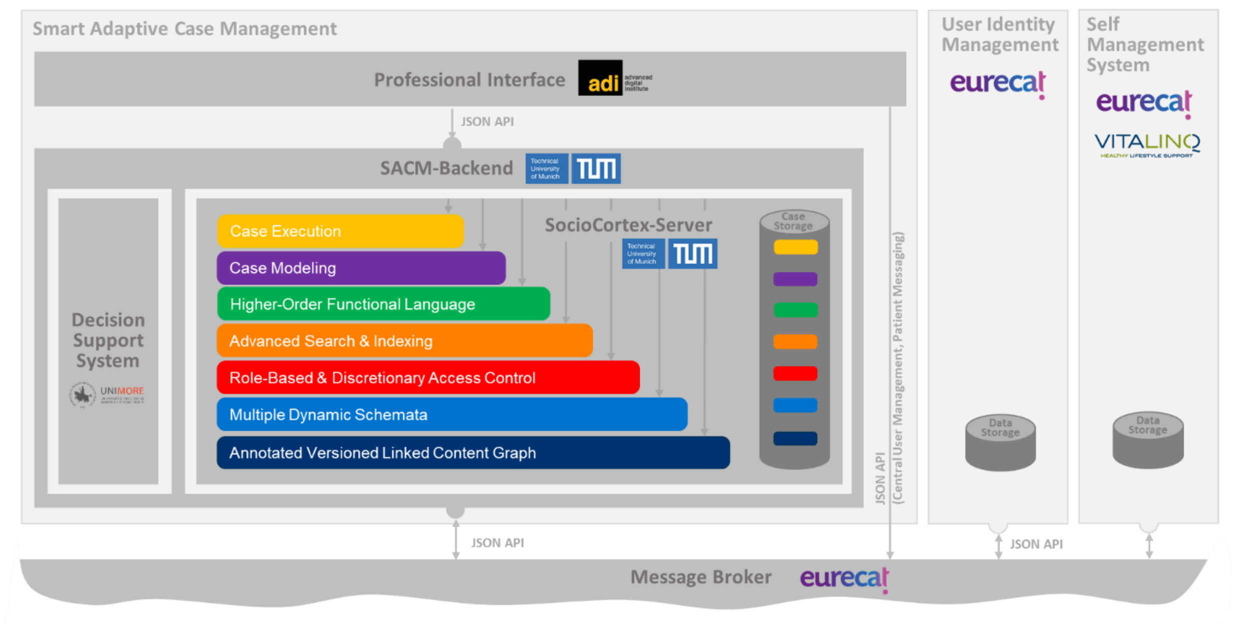


***Figure 1**: Smart Adaptive Case Management Architecture with connected systems including the responsibilities.*

1.1 System Responsibilities

The CONNECARE system consists of several subsystems. To keep duplicated data between the subsystems synchronized, it is relevant to define the leading master system and a slave system. Table 1 illustrates the responsibilities for authentication, user management, user role management and the case authorization across the subsystems.

	User Identity Management	SACM	SMS
Authentication	**Master** (creates token)	**Slave** (validates token)	**Slave** (validates token)
User Management (CRUD Operations)	**Master** (user basic fields)	**Slave** (copy of basic user fields and additional user fields)	**Slave**
User Role Management	**Master**	**Slave**	**Slave**
Case Authorization	**Not Synced** (synced if needed)	**Master**	**Not Synced** (synced if needed)

***Table 1**: System Responsibilities.*

1.2 Authentication (Simple Login)

All necessary login information such as email and password are stored in the User Identity Management to provide a Single Sign On (SSO). Every JWT (JSON Web Token) contains the user id from the User Identity Management and is valid across all subsystems (see Figure 2).

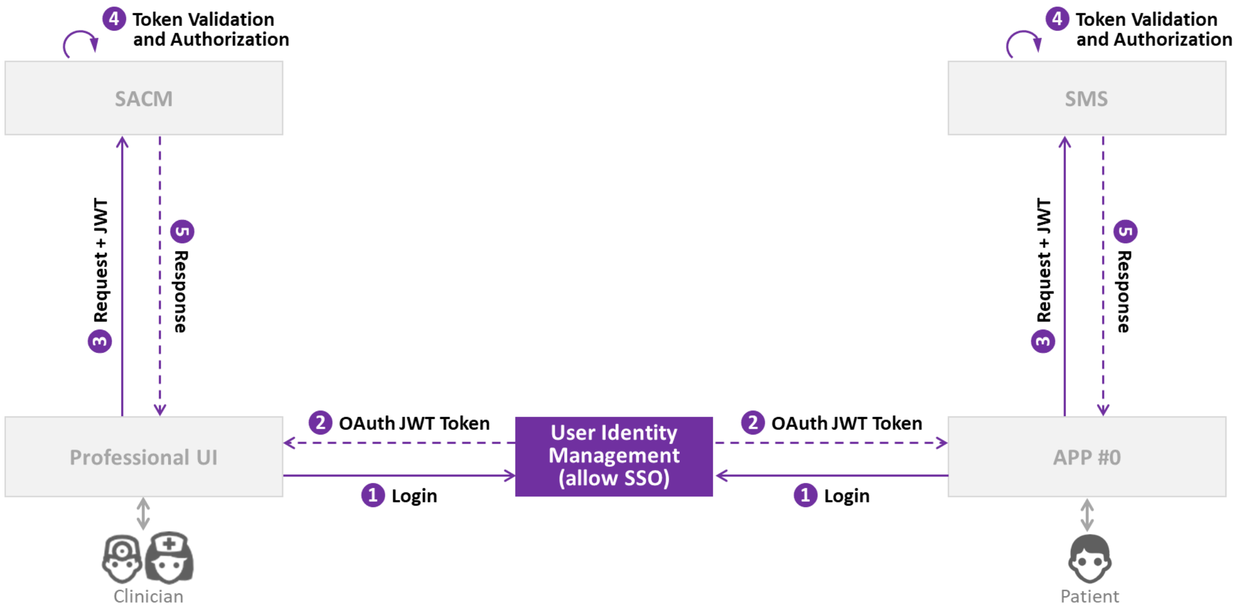


Figure 2: *Authentication via CONNECARE User Identity Management.*

1.3 User Management (CRUD Operations)

Creation of a user on the Professional UI triggers the creation process on the User Identity Management with a UUID that is pushed to the subsystems (see Figure 3). The user creation and deletion follows the same pattern. Additionally, the User Identity Management is responsible for the role management. To integrate the UIM with the SACM, it is necessary that the UIM maps the system unique roles to site specific roles in the SACM. This allows to model access rights with site specific groups.

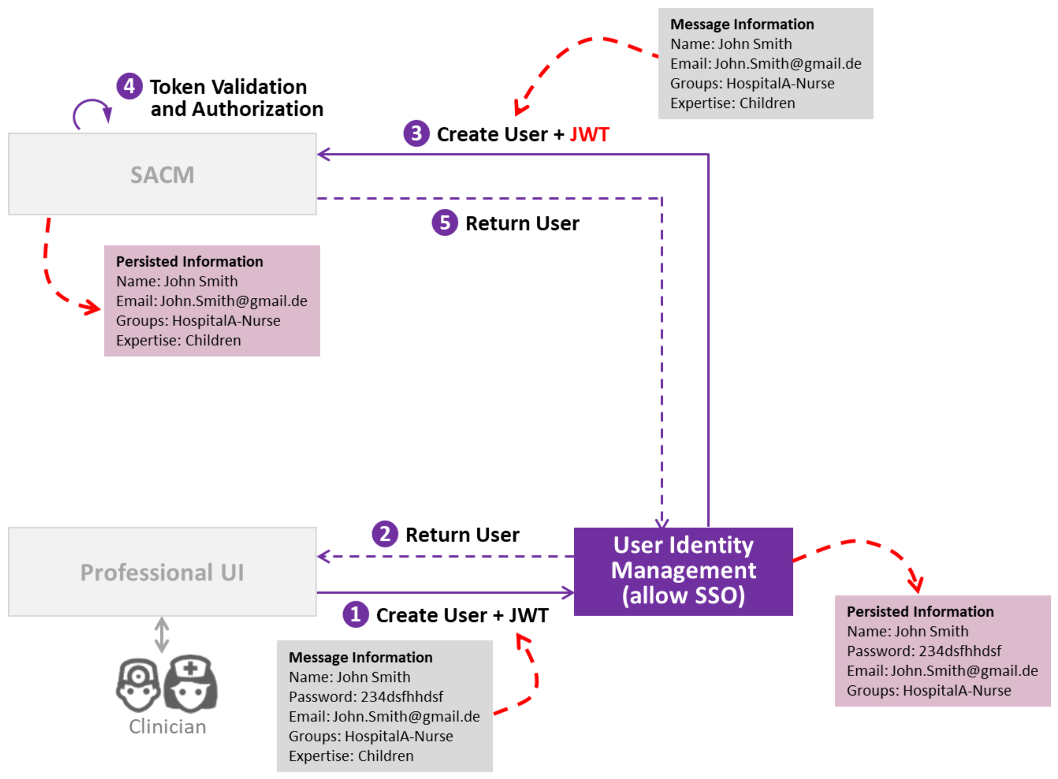


Figure 3: *User creation information flow. The processed and stored information is indicated with grey and red boxes that contain the related user object.*

1.4 Case Authorization (Which user can access a case)

To fulfil the requirements of all the sites, a dedicated access control per case is necessary. All information regarding the case authentication are primarily stored in the SACM backend (see Figure 4). If other subsystems need access this information, an API endpoint can be provided.

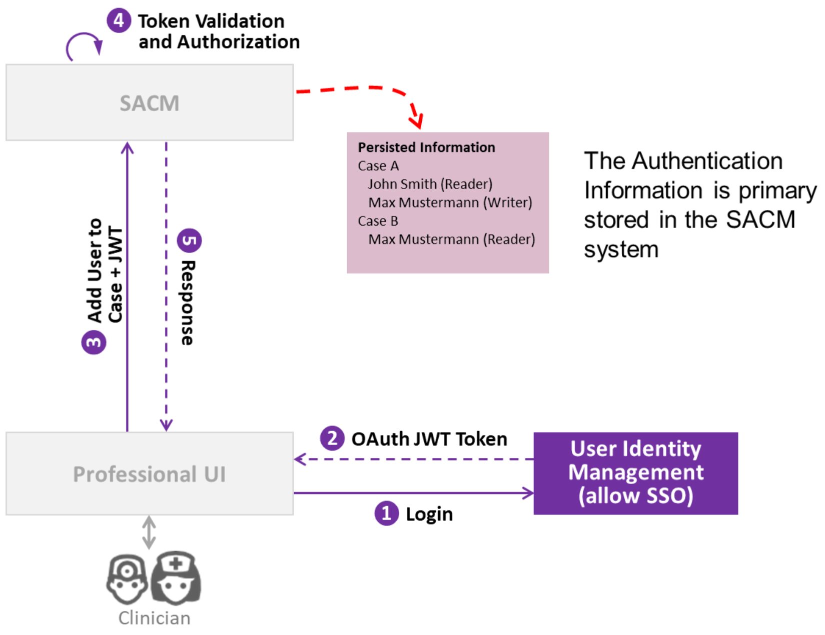


Figure 4: *Case Authorization information flow.*

1.5 SMS Integration

To support execution of the CONNECARE use cases, the SACM system needs to be integrated with the SMS system. The most important use cases that are supported are: 1) *Physical Prescription*, 2) *Patient Monitoring* and 3) *Patient Questionnaires*. Figure 5 illustrates the physical prescription at a conceptual level in CMMN notation. After a clinician has defined the prescription input parameters and completed the first task, the second task is activated. On the activation of the second task, a notification is sent to the SMS system which triggers the execution of the physical prescription (i.e. notification to the patient smartphone/tablet app).

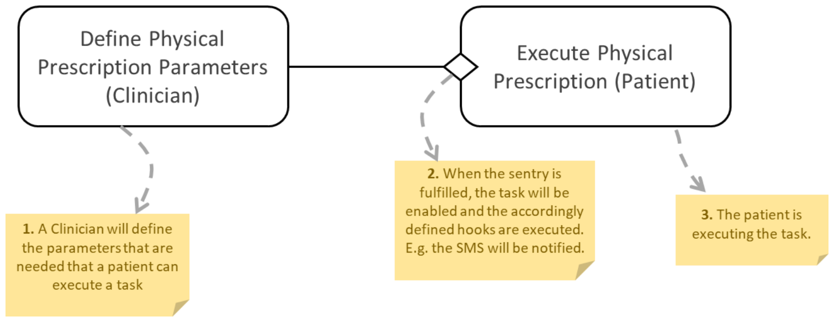

Define Physical Prescription Parameters (Clinician)
Execute Physical Prescription (Patient)
1. A Clinician will define the parameters that are needed that a patient can execute a task
2. When the sentry is fulfilled, the task will be enabled and the accordingly defined hooks are executed. E.g. the SMS will be notified.
3. The patient is executing the task.

Figure 5: *Example physical prescription in CMMN notation, a clinician defines the parameters that are needed for a patient to execute a task.*

Figure 6 illustrates the technical flow on a conceptual level. The precondition is the activation of a process element in this case an *AutomatedTask*. After the activation, a hook notifies the SMS system that a new physical prescription task is available. The patient executes this task via a mobile app or a wearable device. The SMS system can draft the results or directly complete the task in the SACM system.

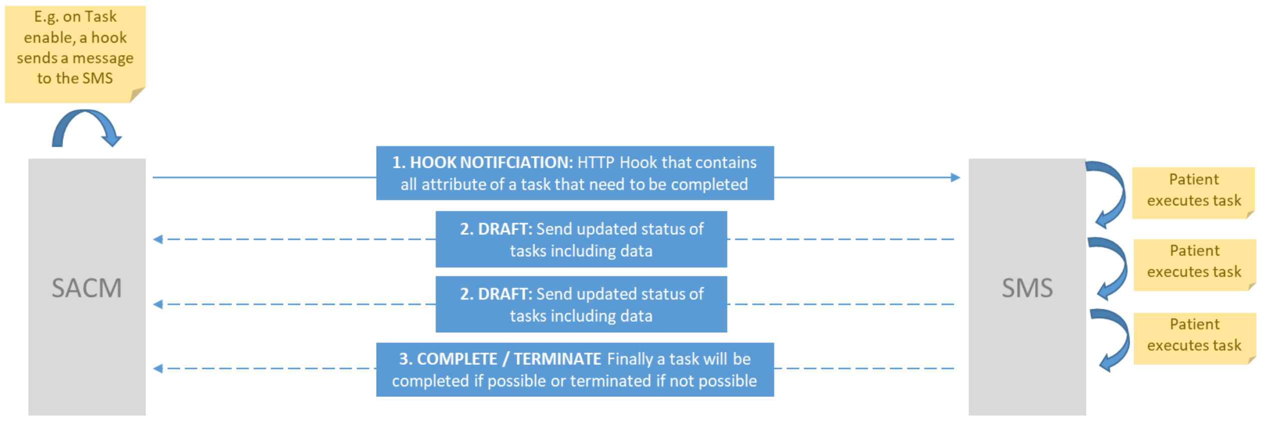
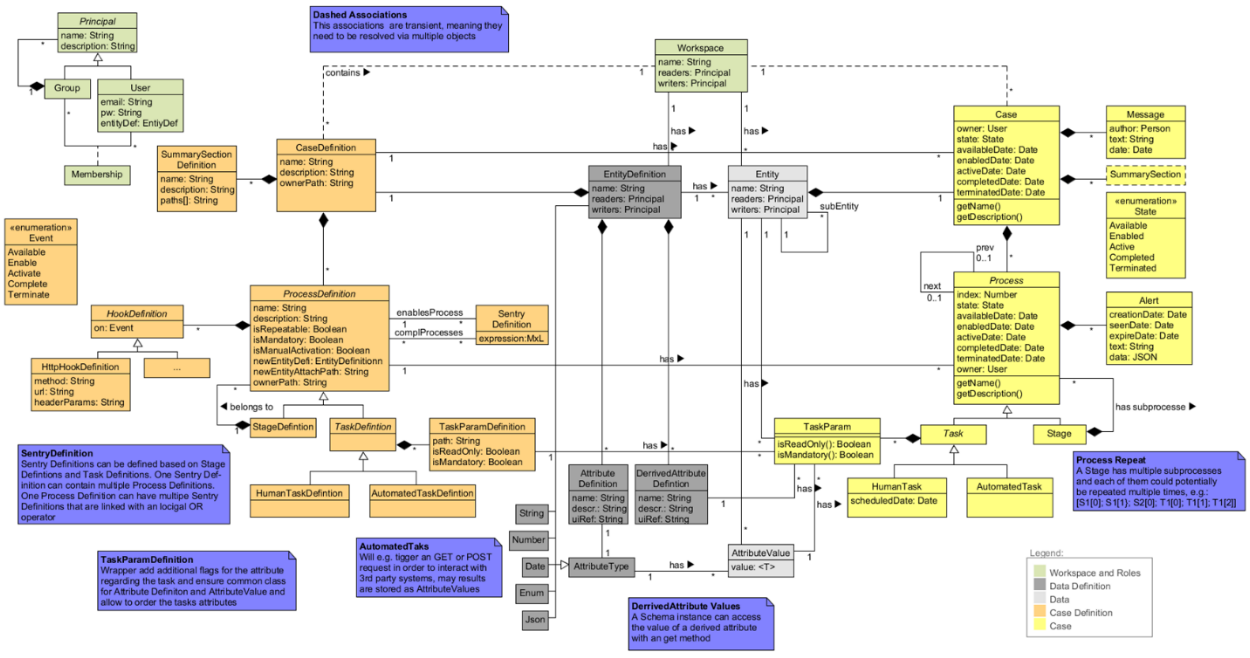

E.g. on Task enable, a hook sends a message to the SMS
SACM
SMS
1. HOOK NOTIFCIATION: HTTP Hook that contains all attribute of a task that need to be completed
2. DRAFT: Send updated status of tasks including data
2. DRAFT: Send updated status of tasks including data
3. COMPLETE / TERMINATE Finally a task will be completed if possible or terminated if not possible
Patient executes task
Patient executes task
Patient executes task

Figure 6*: Conceptual communication flow between SACM and SMS.*

The integration of the other use cases follows the same communication patters with small modifications.

2. Meta-Model

Figure 1 illustrates the SACM meta-model denoted in UML. The UML diagram illustrates six core concepts in different colors. In the center of the diagram, the data definition and data instances are represented in dark and light grey, respectively. The case definition and case instances are indicated in dark and light yellow. The role management is represented on the left and the workspace is represented on the top. In the following sections, the main concepts are explained in more detail.



Dashed Associations
This associations are transient, meaning they need to be resolved via multiple objects

SentryDefinition
Sentry Definitions can be defined based on Stage Defintions and Task Definitions. One Sentry Definition can contain multiple Process Definitions. One Process Definition can have multipe Sentry Definitions that are linked with an locigal OR operator

TaskParamDefinition
Wrapper add additional flags for the attribute regarding the task and ensure common class for Attribute Definiton and AttributeValue and allow to order the tasks attributes

AutomatedTaks
Will e.g. tigger an GET or POST request in order to interact with 3rd party systems, may results are stored as AttributeValues

DerrivedAttribute Values
A Schema instance can access the value of a derived attribute with an get method

Process Repeat
A Stage has multiple subprocesses and each of them could potentially be repeated multiple times, e.g.: [S1[0]; S1[1}; S2[0]; T1[0]; T1[1]; T1[2]]

Figure 7: *Simplified conceptual meta-model.*

2.1 Role Management

The Role Management is considered on a system level, meaning roles work across workspace boundaries.

Principal A principal is an abstraction of a group and a person. This helps to use the concept in a flexible way.

Group A group consist of subgroups and persons who have a membership.

User A user represents a single person in the system. A user can contain multiple attributes.

Membership A membership represents the relationship between user and a group. E.g. the data for creating the membership.

2.2 Workspace

A workspace is a container to store model definitions and the related instances. Each Workspace has read and write Principals. By default, the rights for all contained model element and instances is inherited from their workspace.

2.3 Data Definition

EntityDefinition An entity definition is a container with read and write principals that contains attribute definitions.

AttributeDefinition An attribute definition defines a name and the expected type of the value. Attribute types can be primitive types such as notype, string, longtext, boolean, number, enumeration and date. Additionally, it is possible to reference complex types (groups and users) or custom entity definitions.

DerivedAttributeDefinition A derived attribute is calculated automatically based on other attributes of an entity definition. Derived attribute definitions cannot be nested.

2.4 Data

Entity An entity has inherited read and write principals based on the corresponding workspace and can contain multiple attribute values.

AttributeValue An attribute value contains a value type that is defined in the attribute definition.

2.5 Case Definition

CaseDefinition A case definition is a wrapper container for all process related elements, the data is excluded. Every case definition has a name and provides an owner who is responsible of the instantiated cases. An owner is a user that is derived via a linked attribute definition.

ProcessDefinition A process definition is an abstraction of a stage definition and task definition. Every process definition has a name and contains the information whether the process is mandatory and repeatable. The process definition is a technical abstraction meant to help technical implementation, thus it is used solely at the API level.

SentryDefinition A sentry definition expresses the preconditions of a process definition. One process definition can have multiple sentry definitions that are linked with a logical or. Every sentry definition can consist of several process definitions that are linked with a logical and. A stage or task can only be executed when at least one sentry is fulfilled or no sentry is defined.

StageDefinition A stage definition contains other stage definitions or task definitions.

HumanTaskDefinition A human task definition has multiple task parameters that need to be accomplished to finish the task. Every human task definition has an assigned group that represents the potential task owner during instantiation.

AutomatedTaskDefinition An automated task definition can be used to execute task by a 3rd party system

TaskParmDefinition A task parameter definition represents an attribute definition or derived attribute that belongs to a task and is either in write or read-only state.

HttpHookDefinition An http hook definition defines an HTTP request (GET, PUT, POST, DEL) that is triggered on a certain event (i.e., it is exploited to trigger the SMS and CDSS in this first prototype). For every process definition element events such as AVAILABLE, ENABLE, ACTIVATE, COMPLETE, TERMINATE can be used to define actions.

2.6 Case

Case A case is instantiated based on the related case definition. Every case has a name and an owner represented by a person. The current state is expressed with a state variable and the corresponding dates of state changes tracked.

Process A process is instantiated based on the related process definition. Every process has a name, state and an index that express the current execution iteration. The process is a technical abstraction meant to help technical implementation, thus it is used solely at the API level.

Stage A stage is instantiated based on the related stage definition. Every stage can contain several sub stages or tasks.

HumanTask A human task has an owner that is represented by a user through a linked attribute value. Additionally, a scheduled date expresses when the task should be executed.

AutomatedTask An automated task represents an instantiated automated task definition. Differently with respect to a human task, this needs no input from a user and is executed after the triggering sentry is fulfilled.

2.7 Life Cycle of Process Elements

Every process element has a state that is expressed with the following enumeration values: 1) ***Available,*** which means the model is instantiated, 2) ***Enabled***, which means at least one sentry is fulfilled, i.e. the process could be executed, 3) ***Active,*** which means the process is currently executed, 4a) ***Completed,*** which means the process was successfully completed, 4b) ***Terminated,*** which means the process was aborted and therefore not successfully completed.

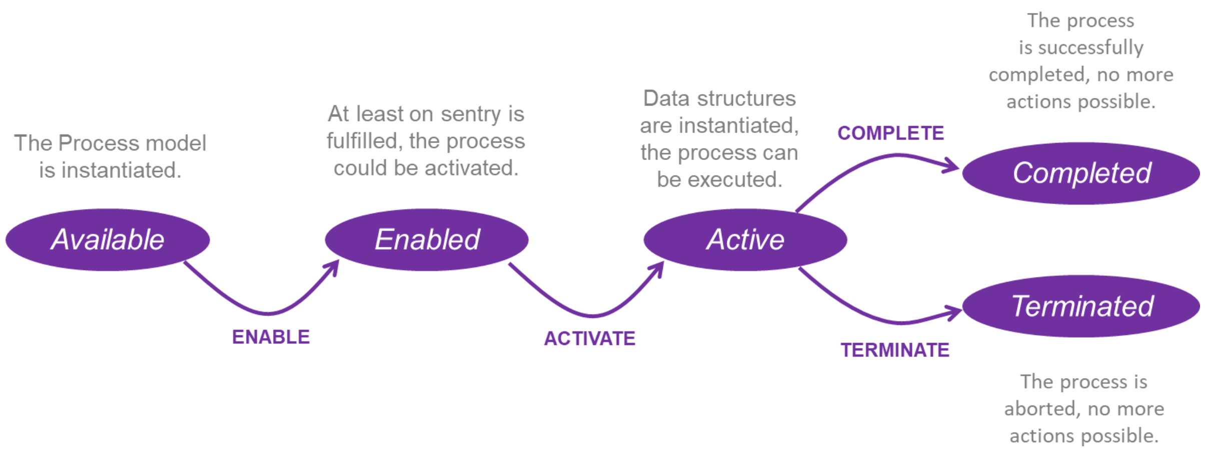

The Process model is instantiated.
At least on sentry is fulfilled, the process could be activated.
Data structures are instantiated, the process can be executed.
The process is successfully completed, no more actions possible.
COMPLETE
Completed
Available
Enabled
Active
Terminated
ENABLE
ACTIVATE
TERMINATE
The process is aborted, no more actions possible.

Figure 8: *SACM possible process states.*

2.8 Process State Change Notifications

All process elements support notifications based on process state change events. State change events are: *1) available, 2) enable, 3) activate, 4) complete, 5) terminate*. All hooks are executed after the state change event has been applied. A hook represents basically the following http request methods: POST, GET, PUT, DELETE. The hook body contains by default the JSON serialized process element.

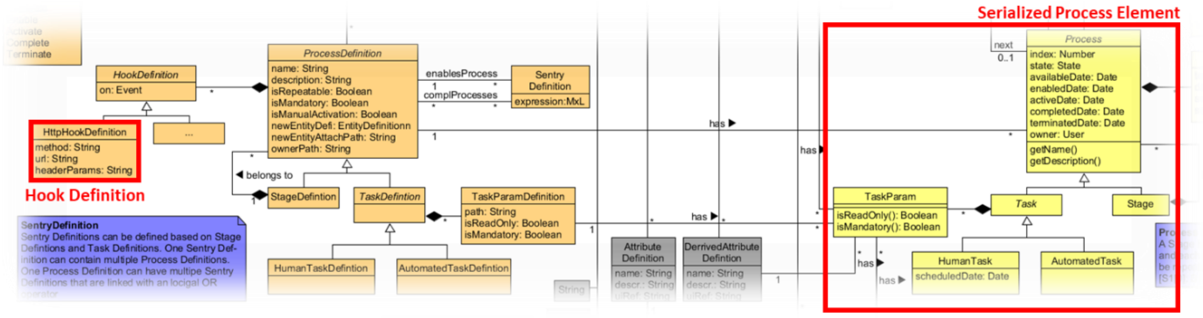


Figure 9: *SACM Meta-Model with highlighted HttpHookDefinition and Serialized Process Element.*

2.9 Process Alerts

All process elements provide alerts to notify the clinician in case of anomalies. In general, there are two different scenarios supported: First, a *HumanTask* is not completed on the scheduled date and the clinician needs to be remembered to complete this task. Alternately, *AutomatedTasks* that are executed by third party systems are providing values that are above or below a certain threshold value. This way, third party systems may create alerts depending on their own logic by exploiting automated tasks.

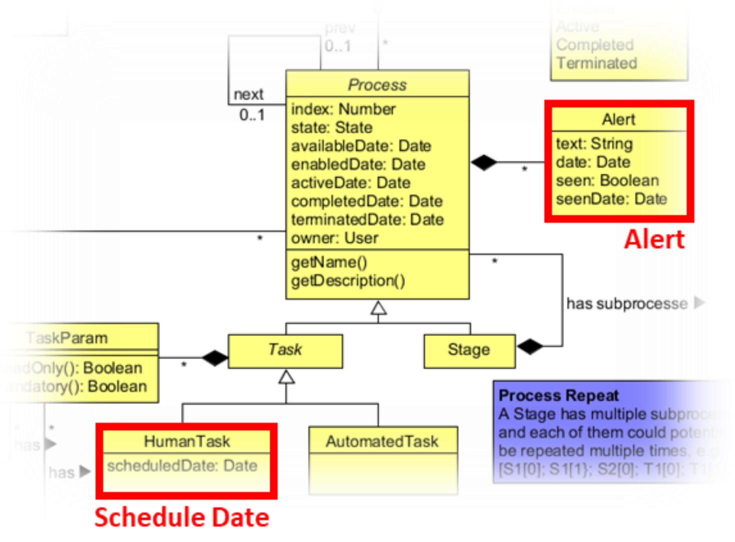


***Figure 10:** SACM Meta-Model with highlighted alert.*

2.10 Third party UUIDs as primary IDs for model Elements

The SACM system is integrated with different third party systems such as the User Identity Management and the SMS. To simplify the integration process, the SACM system supports to create many model elements with foreign IDs. The SACM system accepts IDs with up to 32, UTF-8 formatted, characters. The attempt to create a model element with an already existing ID leads to an exception. This foreign ID pattern is used, e.g. for the user and group creation with UUIDs form the UIM.

2.11 Model based Language Support

The SACM system will be used on different sites that require local language support. All relevant model elements support definition of a technical English name for maintainability and a displayed name in the related local language for the UI. In the frontend, the translation is only applied for the static UI element names but not for model based elements. This increases the systems maintainability due to the fact the model contains all necessary information. Currently the translation is limited to one additional language.

2.12 Complex Sentries Preconditions

The SACM meta-model supports complex sentry definitions for every process definition. A process definition can be either a *Stage* definition, *HumanTask* definition or an *AutomatedTask* definition. One process definition can have multiple sentry definitions that are linked via a logical *or*. One Sentry definition can contain multiple preconditions that are linked with a logical *and*. A simple precondition is that a certain process needs to be completed. A complex precondition is expressed with our model based query language mxl[2]. For example, a certain value of a task need to be greater than another value.

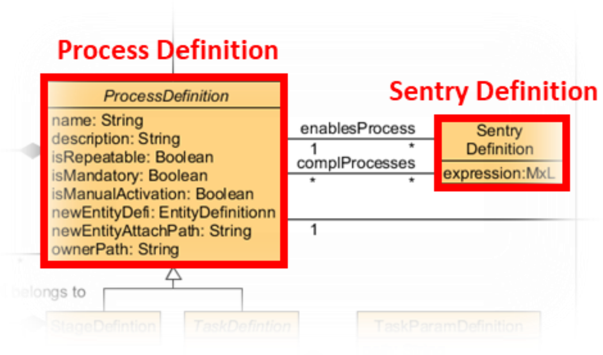


Figure 11: *SACM Meta-Model with highlighted sentry definition.*

```
(number(lace3,0)+number(lace4,0)+number(lace5,0)+number(lace6,0)+number(lace7,0)+numb
er(lace8,0)+number(lace9,0)+number(lace10,0)+number(lace11,0)+number(lace12,0)+number
(lace13,0)+number(lace14,0)+number(lace15,0)+number(lace16,0)) > 5 OR lace3<2
```

Figure 12: *Sample mxl expression based on the lace questionnaire.*

[2] http://www.sociocortex.com/tutorial/2015/12/01/mxl01/

2.13 Data storage and Execution semantics separation

The SACM meta-model clearly separates the data storage from the execution semantics (see Figure 13). Multiple reasons motivate this architectural choice: 1) data that is initially created based on one task can be used in other tasks to show or review the data, 2) external data sources can be easily integrated without being necessarily part of a process, 3) in general, the data objects are probably more interesting in the long run while the execution history may be deleted after its exertion.

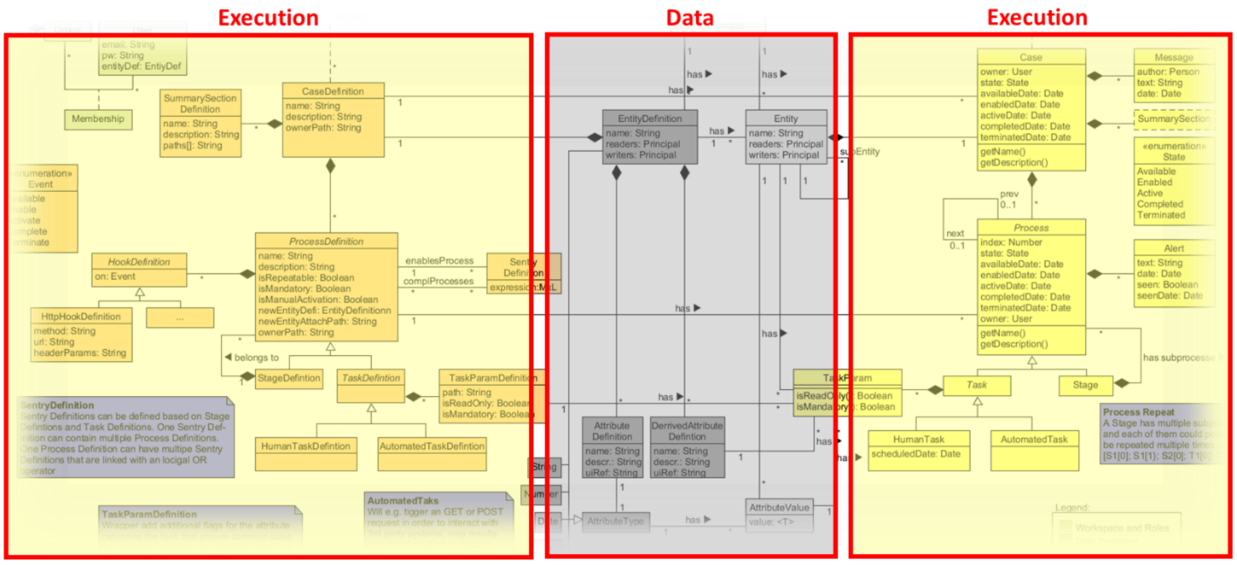


Figure 13: *SACM Meta-Model with highlighted separation of the data storage and the execution semantics.*

2.14 Historical Development of the Meta-Model

The SACM Meta-Model builds on the experiences made with SocioCortex. The system has a long history and has proven to be very reliable and stable as a productive system.

2.14.1 SocioCortex - Historical Version

SocioCortex uses several model-based concepts to structure knowledge according to specific needs (see Figure 1). A Workspace represents a container that contains multiple model elements with certain access rights such as read or write. Entity definitions are structuring elements that are used to bundle a set of attribute definitions or derived attribute definitions. An attribute has a certain type that can be either a complex type such as an entity definition or a primitive type such as a string, number, etc. These definitions can be instantiated, causing a related entity with their attributes and derived attributes to be created. By default, Entities themselves have inherited access rights from the workspace that can be overwritten with dedicated readers. The access right management supports complex use cases based on the concepts groups and users. The group and the user extend the abstract principal to support nesting based on a composite pattern.

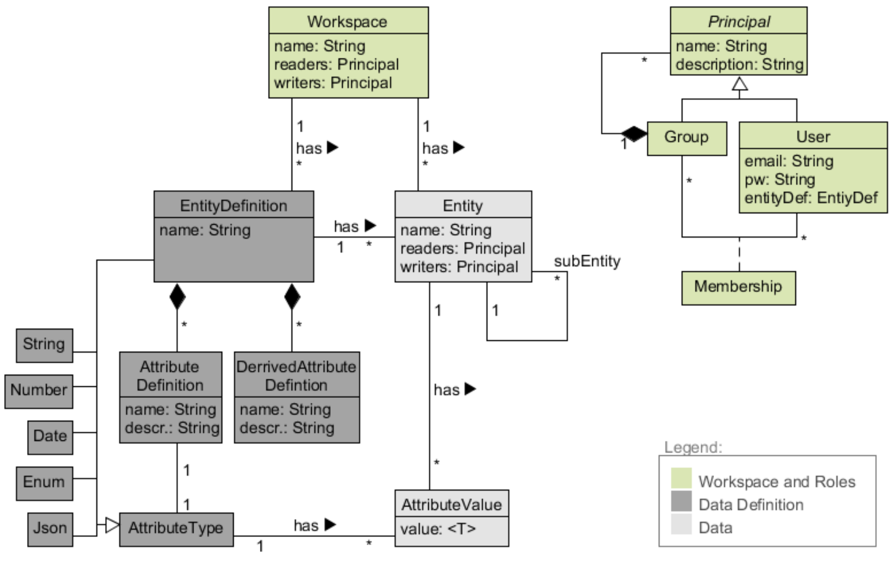


Figure 14: *Simplified conceptual meta-model of SocioCortex (version 2014, naming adapted) based on (Matthes, 2011).*

2.14.2 SocioCortex - Version 2015

Wiki pages are extended with Tasks, Attributes, and a Type to structure knowledge-intensive processes in work plans. Additional concepts for *TaskDefinition*, *TypeDefinition*, and *AttributeDefinition* are used to specify integrity constraints for predefined work templates (see Figure 15). Work plans are structured by end-users, while work templates are defined by modeling experts. A type definition is loosely coupled to a type to enable the creation of new types by end-users on the fly at run-time. The same applies to attribute definition and attribute, as well as, task definition and task. Tasks are mapped either to *HumanTasks* or *CaseTasks* in CMMN depending on the assigned Entities. Stages are concepts that serve as containers for *TaskDefinitions* and track the behavior of a wiki page at run-time. The *TypeDefinition* is a composite that facilitates the modeling of complex hierarchical information and task structures. Tasks that reference other pages are automatically finished after all subtasks on these referenced pages are completed. Sentries specify logical entry and exit criteria between tasks and stages, e.g. a task is only enabled after certain other tasks are completed. (Matheus Hauder, 2015).

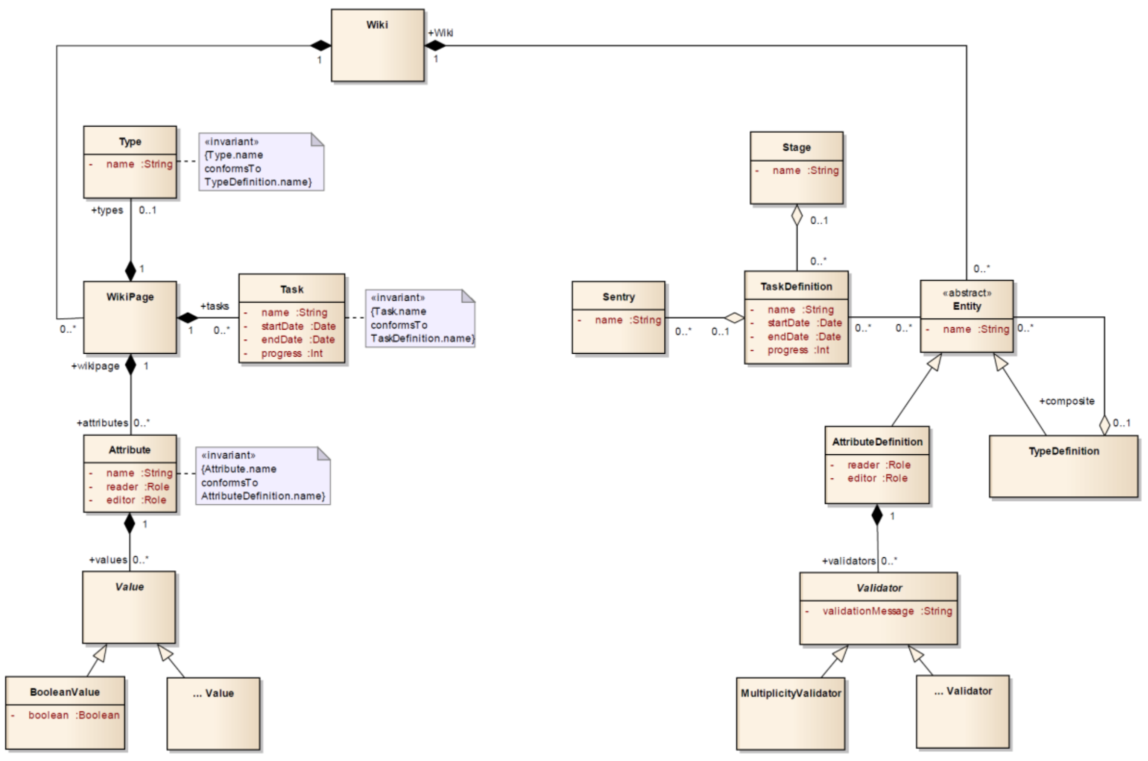


Figure 15: *Simplified conceptual meta-model of the SocioCortex Wiki showing the core elements to support the collaborative structuring of processes for knowledge work adapted from (Matheus Hauder, 2015).*

2.14.3 Comparison with SACM Meta-Model

The CONNECARE SACM-Backend is developed based on the SocioCortex core (version 2015). Therefore, the SocioCortex functionalities are mainly extended on three conceptual layers: *1) support a sustainable user and access right management, 2) case modelling and 3) case execution*. CONNECARE specific functional and non-functional requirements lead to major changes (see Figure 16).

	SocioCortex (version 2015)	SACM-Backend
Case Execution	**(Yes)**	**Yes**
Support overdue alerts that expire	(Yes)	Yes
Support third party alerts that expire	No	Yes
Support case bases messaging	No	Yes
Support explicit case and process states	No	Yes
Case Modeling	**(Yes)**	**Yes**
Supports stages and tasks with sentries as preconditions	Yes	Yes
Supports automated tasks	No	Yes
Support expressions as part of the sentries	No	Yes
Support separation of data storage and execution semantics	No	Yes
Support third party system integration (hooks)	No	Yes
Support manual activation tasks	No	Yes
Support mandatory and optional task parameters	No	Yes
Support read only and writable task parameters	No	Yes
Support case summary sections	No	Yes
Higher-Order Functional Language	**Yes**	**Yes**
Support string to number transformation	Yes	Yes
Advanced Search & Indexing	**Yes**	**Yes**
Role-Based & Discretionary Access Control	**Yes**	**Yes**
Support external user identity management	No	Yes
Support foreign user and group ids	(Yes)	Yes
Support custom user model	Yes	Yes
Multiple Dynamic Schemata	**(Yes)**	**Yes**
Support enumerations with key value pairs	No	Yes
Support schema based validation on API	No	Yes
Support user referencing constraint - e.g. specific group	No	Yes
Annotated Versioned Linked Content Graph	**Yes**	**Yes**

Figure 16: *Meta-Model comparison according to CONNECARE requirements. As baseline SocioCortex form 2015 is used and compared with the current SACM-Backend prototype that uses the latest version of SocioCortex as core.*

3. Case Model Definition

3.1 Identified User Groups of each Site

The defined roles are the first step to model cases in the SACM system. The user groups are site specific, Table 2 illustrates all identified roles of each site.

	Barcelona			Lleida		ASSUTA		Groningen		
Role	**CS1**	**CS2**	**CS3**	**CS1**	**CS2**	**CS1**	**CS2**	**CS1 - embrace**	**CS1 - AC service**	**CS2**
Patient	F	F	F	F	F	F	F	F	F	F
Clinician	A-D	A-D	A-D	A-E	A-E	A-D	A-D		A-B	A-D
Case Manager	A-D	A-D	A-D	A-E	A-E	A-D	A-D	A-D	A-D	A-D
Nurse	A-B	A-C	A-C	E	E	A-B	A-B			A-B
Anesthesiologist		A-D	A-D		A-B	A-B	A-B			
Administrative officers	A									
Physician						A-B	A-B		A-B	A-B
Physiotherapist		A-C	A-C		A-B	A-B	A-B			
Psychologist		A-B	A-B							
Nutritionist		A-B	A-B							
Social worker				A-B	A-B	A-B	A-B	A-B		
Primary Care Clinician					A-B	A-B	A-B			
Lab technician									A-B	
Local pulmonologists									A-B	
Data manager	A	A	A					A-B		
Secretary / Typist						A-D	A-D			
Site Adminstrator	E	E	E			E	E	E	E	E
Researcher	A	A	A			E	E	A-B		

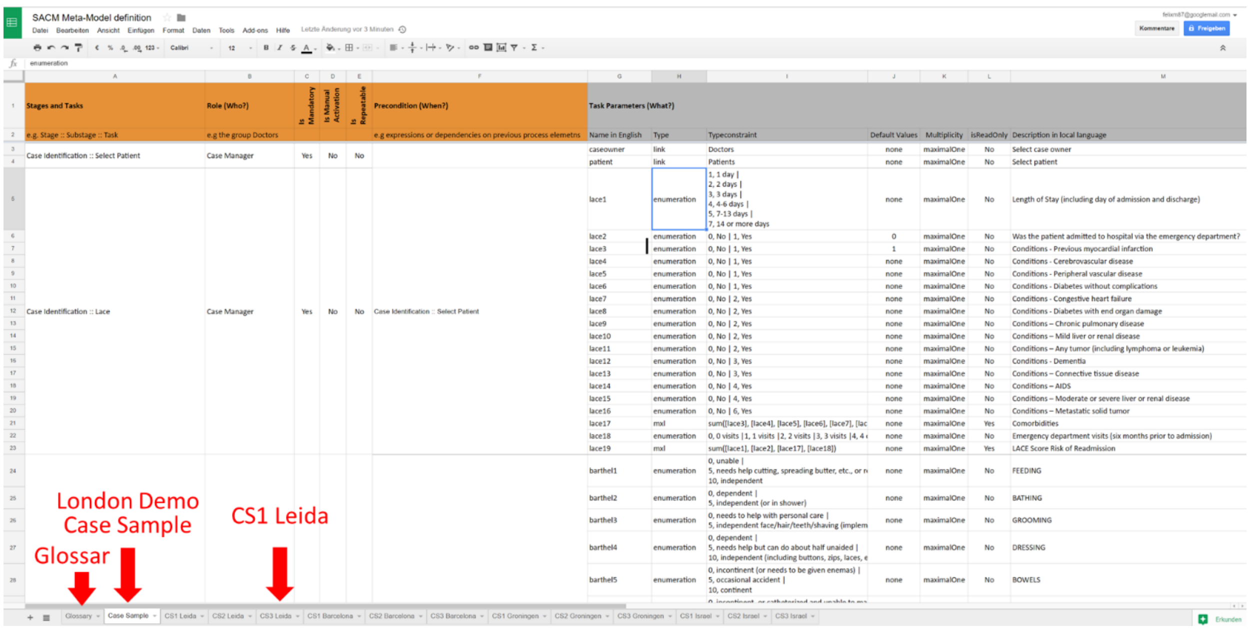
Access Levels

A) Case Viewer (Read Only Access)
B) Case Worker (Complete Tasks)
C) Case Owner (Create, Read, Update, Delete)
D) Manage Patients (Create, Read, Update, Delete)
E) Manage Professionals (Create, Read, Update, Delete)
F) Patient (only used for case assignment - no read access)

***Table 2**: User Groups of each site.*

3.2 Transformation of conceptual CMMN models into executable SACM Models

All clinical sites documented their workflow based on CMMN diagrams. Additional information regarding the necessary attributes of every task are documented with tables embedded in the requirements document. However, this conceptual requirement documents lack some detailed information, e.g. the exact preconditions of task, the attributes multiplicities, default values etc. Therefore, additional enhancement of the information was necessary. For collecting this information, a Google spreadsheet was created and shared with all clinical partners. The spreadsheet consists of a glossary tab that explains the relevant model elements, a sample tab that illustrates a fully modelled case sample and one tab for each site and case study.

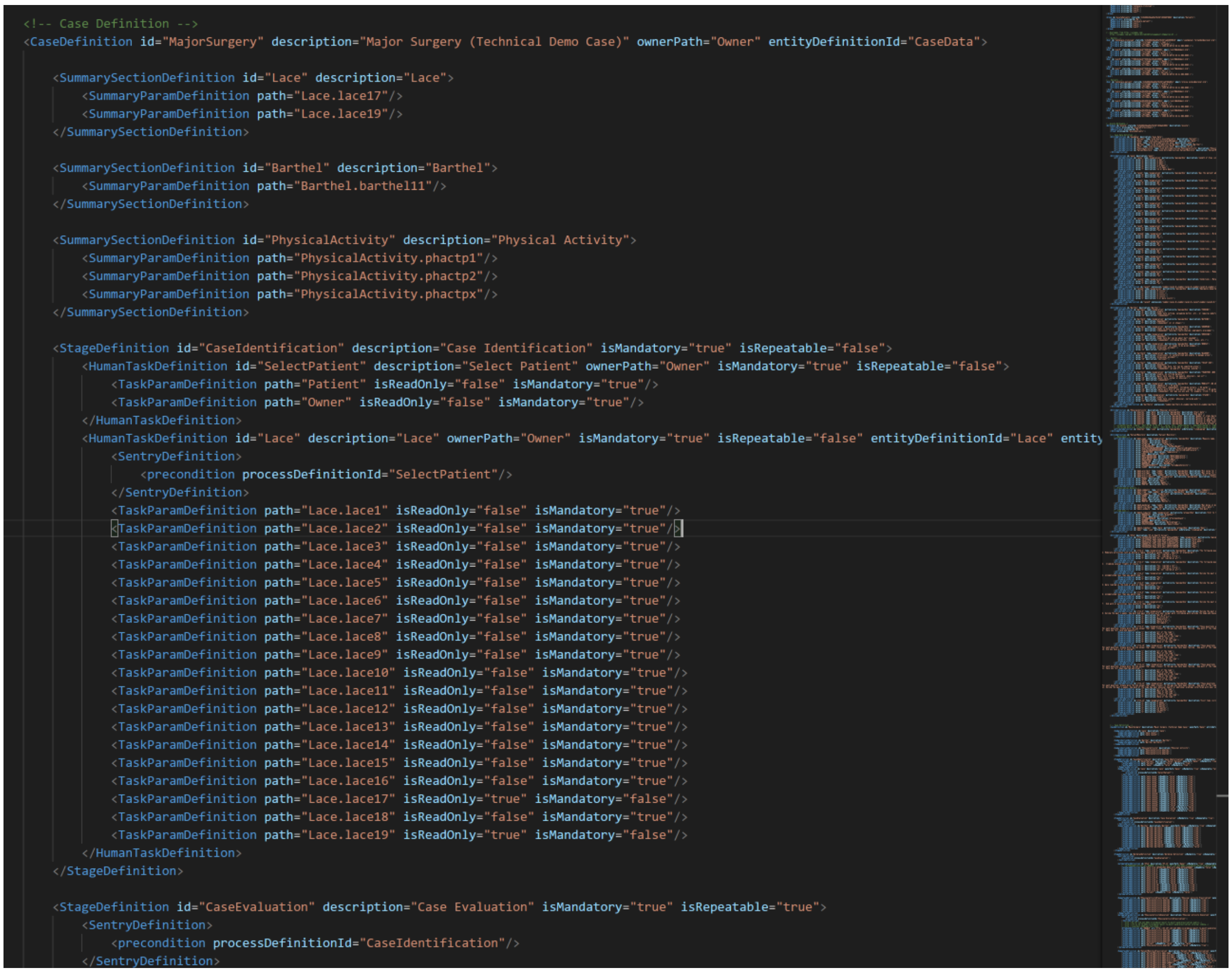


	A	B	C	D	E	F	G	H	I	J	K	L	M
1	Stages and Tasks	Role (Who?)	Is Mandatory	Is Manual Activation	Is Repeatable	Precondition (When?)	Task Parameters (What?)						
2	e.g. Stage :: Substage :: Task	e.g the group Doctors				e.g expressions or dependencies on previous process elemetns	Name in English	Type	Typeconstraint	Default Values	Multiplicity	isReadOnly	Description in local language
3	Case Identification :: Select Patient	Case Manager	Yes	No	No		caseowner	link	Doctors	none	maximalOne	No	Select case owner
4							patient	link	Patients	none	maximalOne	No	Select patient
5	Case Identification :: Lace	Case Manager	Yes	No	No	Case Identification :: Select Patient	lace1	enumeration	1, 1 day | 2, 2 days | 3, 3 days | 4, 4-6 days | 5, 7-13 days | 7, 14 or more days	none	maximalOne	No	Length of Stay (including day of admission and discharge)
6							lace2	enumeration	0, No | 1, Yes	0	maximalOne	No	Was the patient admitted to hospital via the emergency department?
7							lace3	enumeration	0, No | 1, Yes	1	maximalOne	No	Conditions - Previous myocardial infarction
8							lace4	enumeration	0, No | 1, Yes	none	maximalOne	No	Conditions - Cerebrovascular disease
9							lace5	enumeration	0, No | 1, Yes	none	maximalOne	No	Conditions - Peripheral vascular disease
10							lace6	enumeration	0, No | 1, Yes	none	maximalOne	No	Conditions - Diabetes without complications
11							lace7	enumeration	0, No | 2, Yes	none	maximalOne	No	Conditions - Congestive heart failure
12							lace8	enumeration	0, No | 2, Yes	none	maximalOne	No	Conditions - Diabetes with end organ damage
13							lace9	enumeration	0, No | 2, Yes	none	maximalOne	No	Conditions – Chronic pulmonary disease
14							lace10	enumeration	0, No | 2, Yes	none	maximalOne	No	Conditions – Mild liver or renal disease
15							lace11	enumeration	0, No | 2, Yes	none	maximalOne	No	Conditions – Any tumor (including lymphoma or leukemia)
16							lace12	enumeration	0, No | 3, Yes	none	maximalOne	No	Conditions - Dementia
17							lace13	enumeration	0, No | 3, Yes	none	maximalOne	No	Conditions – Connective tissue disease
18							lace14	enumeration	0, No | 4, Yes	none	maximalOne	No	Conditions – AIDS
19							lace15	enumeration	0, No | 4, Yes	none	maximalOne	No	Conditions – Moderate or severe liver or renal disease
20							lace16	enumeration	0, No | 6, Yes	none	maximalOne	No	Conditions – Metastatic solid tumor
21							lace17	mxl	sum([lace3], [lace4], [lace5], [lace6], [lace7], [lac	none	maximalOne	Yes	Comorbidities
22							lace18	enumeration	0, 0 visits |1, 1 visits |2, 2 visits |3, 3 visits |4, 4	none	maximalOne	No	Emergency department visits (six months prior to admission)
23							lace19	mxl	sum([lace1], [lace2], [lace17], [lace18])	none	maximalOne	Yes	LACE Score Risk of Readmission
24							barthel1	enumeration	0, unable | 5, needs help cutting, spreading butter, etc., or r 10, independent	none	maximalOne	No	FEEDING
25							barthel2	enumeration	0, dependent | 5, independent (or in shower)	none	maximalOne	No	BATHING
26							barthel3	enumeration	0, needs to help with personal care | 5, independent face/hair/teeth/shaving (implem	none	maximalOne	No	GROOMING
27							barthel4	enumeration	0, dependent | 5, needs help but can do about half unaided | 10, independent (including buttons, zips, laces, e	none	maximalOne	No	DRESSING
28							barthel5	enumeration	0, incontinent (or needs to be given enemas) | 5, occasional accident | 10, continent	none	maximalOne	No	BOWELS

Figure 17: *Google Spreadsheet to define all conceptual details of every case study on all sites.*

3.3 XML Case Definition File

To transform the descriptive spreadsheet models into executable cases an XML case definition file is created. Figure 1 illustrates a sample case definition that defines the case, as well as, the groups and the initial set of users. After the case is defined via XML the file is imported an executable model definition is created.

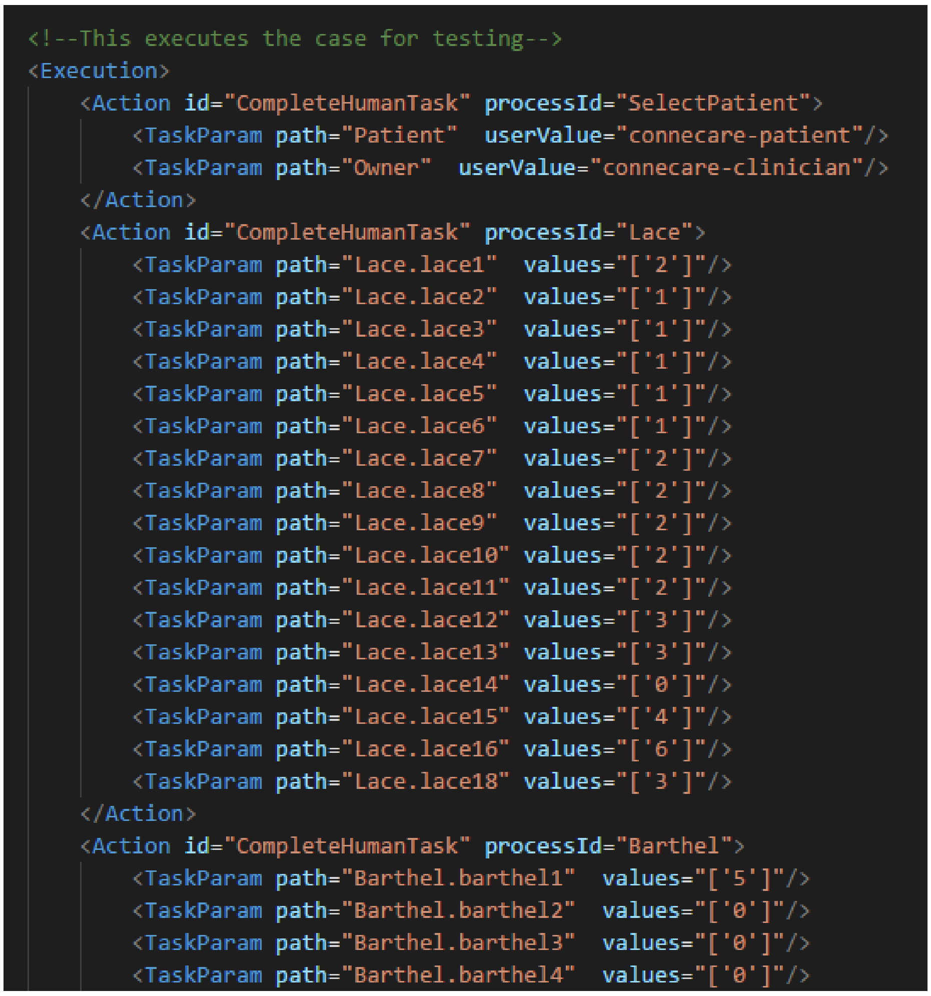


```xml
<!-- Case Definition -->
<CaseDefinition id="MajorSurgery" description="Major Surgery (Technical Demo Case)" ownerPath="Owner" entityDefinitionId="CaseData">

    <SummarySectionDefinition id="Lace" description="Lace">
        <SummaryParamDefinition path="Lace.lace17"/>
        <SummaryParamDefinition path="Lace.lace19"/>
    </SummarySectionDefinition>

    <SummarySectionDefinition id="Barthel" description="Barthel">
        <SummaryParamDefinition path="Barthel.barthel11"/>
    </SummarySectionDefinition>

    <SummarySectionDefinition id="PhysicalActivity" description="Physical Activity">
        <SummaryParamDefinition path="PhysicalActivity.phactp1"/>
        <SummaryParamDefinition path="PhysicalActivity.phactp2"/>
        <SummaryParamDefinition path="PhysicalActivity.phactpx"/>
    </SummarySectionDefinition>

    <StageDefinition id="CaseIdentification" description="Case Identification" isMandatory="true" isRepeatable="false">
        <HumanTaskDefinition id="SelectPatient" description="Select Patient" ownerPath="Owner" isMandatory="true" isRepeatable="false">
            <TaskParamDefinition path="Patient" isReadOnly="false" isMandatory="true"/>
            <TaskParamDefinition path="Owner" isReadOnly="false" isMandatory="true"/>
        </HumanTaskDefinition>
        <HumanTaskDefinition id="Lace" description="Lace" ownerPath="Owner" isMandatory="true" isRepeatable="false" entityDefinitionId="Lace" entity
            <SentryDefinition>
                <precondition processDefinitionId="SelectPatient"/>
            </SentryDefinition>
            <TaskParamDefinition path="Lace.lace1" isReadOnly="false" isMandatory="true"/>
            <TaskParamDefinition path="Lace.lace2" isReadOnly="false" isMandatory="true"/>
            <TaskParamDefinition path="Lace.lace3" isReadOnly="false" isMandatory="true"/>
            <TaskParamDefinition path="Lace.lace4" isReadOnly="false" isMandatory="true"/>
            <TaskParamDefinition path="Lace.lace5" isReadOnly="false" isMandatory="true"/>
            <TaskParamDefinition path="Lace.lace6" isReadOnly="false" isMandatory="true"/>
            <TaskParamDefinition path="Lace.lace7" isReadOnly="false" isMandatory="true"/>
            <TaskParamDefinition path="Lace.lace8" isReadOnly="false" isMandatory="true"/>
            <TaskParamDefinition path="Lace.lace9" isReadOnly="false" isMandatory="true"/>
            <TaskParamDefinition path="Lace.lace10" isReadOnly="false" isMandatory="true"/>
            <TaskParamDefinition path="Lace.lace11" isReadOnly="false" isMandatory="true"/>
            <TaskParamDefinition path="Lace.lace12" isReadOnly="false" isMandatory="true"/>
            <TaskParamDefinition path="Lace.lace13" isReadOnly="false" isMandatory="true"/>
            <TaskParamDefinition path="Lace.lace14" isReadOnly="false" isMandatory="true"/>
            <TaskParamDefinition path="Lace.lace15" isReadOnly="false" isMandatory="true"/>
            <TaskParamDefinition path="Lace.lace16" isReadOnly="false" isMandatory="true"/>
            <TaskParamDefinition path="Lace.lace17" isReadOnly="true" isMandatory="false"/>
            <TaskParamDefinition path="Lace.lace18" isReadOnly="false" isMandatory="true"/>
            <TaskParamDefinition path="Lace.lace19" isReadOnly="true" isMandatory="false"/>
        </HumanTaskDefinition>
    </StageDefinition>

    <StageDefinition id="CaseEvaluation" description="Case Evaluation" isMandatory="true" isRepeatable="true">
        <SentryDefinition>
            <precondition processDefinitionId="CaseIdentification"/>
        </SentryDefinition>
```

***Figure 18**: XML Case Definition File.*

3.4 XML Case Definition File (Testing)

Testing the complex case models simply via API is not sufficient. Therefore, the XML case definition file allows to define one sample execution of a case (see Figure 19). The import script can be used with an additional flag that triggers the execution with a defined user. After the execution, the case data is not deleted to allow viewing the partly or fully executed case via Professional User Interface.

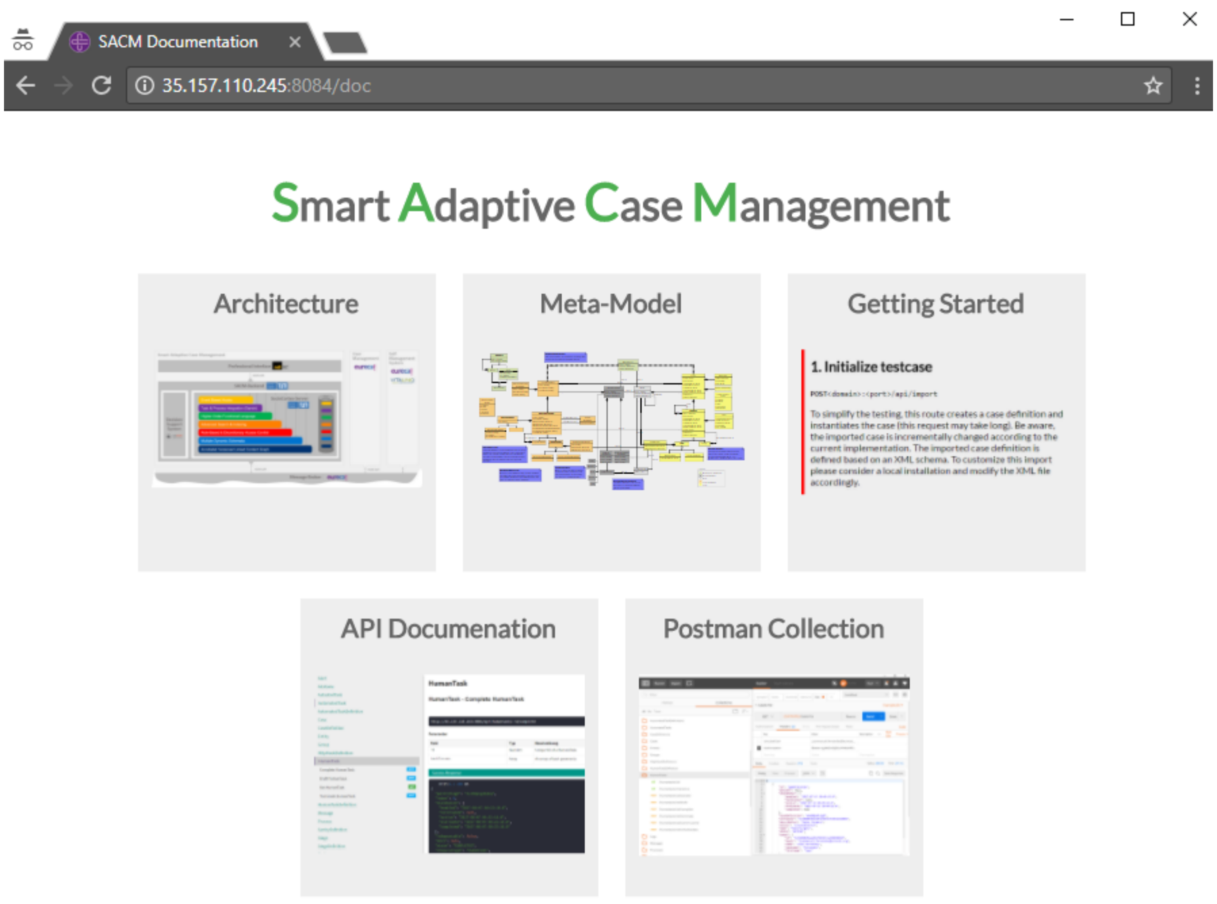


```xml
<!--This executes the case for testing-->
<Execution>
    <Action id="CompleteHumanTask" processId="SelectPatient">
        <TaskParam path="Patient"  userValue="connecare-patient"/>
        <TaskParam path="Owner"  userValue="connecare-clinician"/>
    </Action>
    <Action id="CompleteHumanTask" processId="Lace">
        <TaskParam path="Lace.lace1"  values="['2']"/>
        <TaskParam path="Lace.lace2"  values="['1']"/>
        <TaskParam path="Lace.lace3"  values="['1']"/>
        <TaskParam path="Lace.lace4"  values="['1']"/>
        <TaskParam path="Lace.lace5"  values="['1']"/>
        <TaskParam path="Lace.lace6"  values="['1']"/>
        <TaskParam path="Lace.lace7"  values="['2']"/>
        <TaskParam path="Lace.lace8"  values="['2']"/>
        <TaskParam path="Lace.lace9"  values="['2']"/>
        <TaskParam path="Lace.lace10" values="['2']"/>
        <TaskParam path="Lace.lace11" values="['2']"/>
        <TaskParam path="Lace.lace12" values="['3']"/>
        <TaskParam path="Lace.lace13" values="['3']"/>
        <TaskParam path="Lace.lace14" values="['0']"/>
        <TaskParam path="Lace.lace15" values="['4']"/>
        <TaskParam path="Lace.lace16" values="['6']"/>
        <TaskParam path="Lace.lace18" values="['3']"/>
    </Action>
    <Action id="CompleteHumanTask" processId="Barthel">
        <TaskParam path="Barthel.barthel1"  values="['5']"/>
        <TaskParam path="Barthel.barthel2"  values="['0']"/>
        <TaskParam path="Barthel.barthel3"  values="['0']"/>
        <TaskParam path="Barthel.barthel4"  values="['0']"/>
```

***Figure 19**: XML Case Definition File – Test Definition*

4. Documentation

All relevant technical documentation regarding the SACM system and its components is made available on a webpage[3] (see Figure 20). The documentation contains: 1) the *architecture* description, 2) the *meta-model* with explanation of the main elements, 3) a *getting started* tutorial that explains how to start development with SocioCortex, 4) an API documentation that documents the available resources and 5) additionally a postman collection to easily test the API endpoints. To ensure that the documentation is aligned with the code, most of the documentation is contained inside the GIT repository.



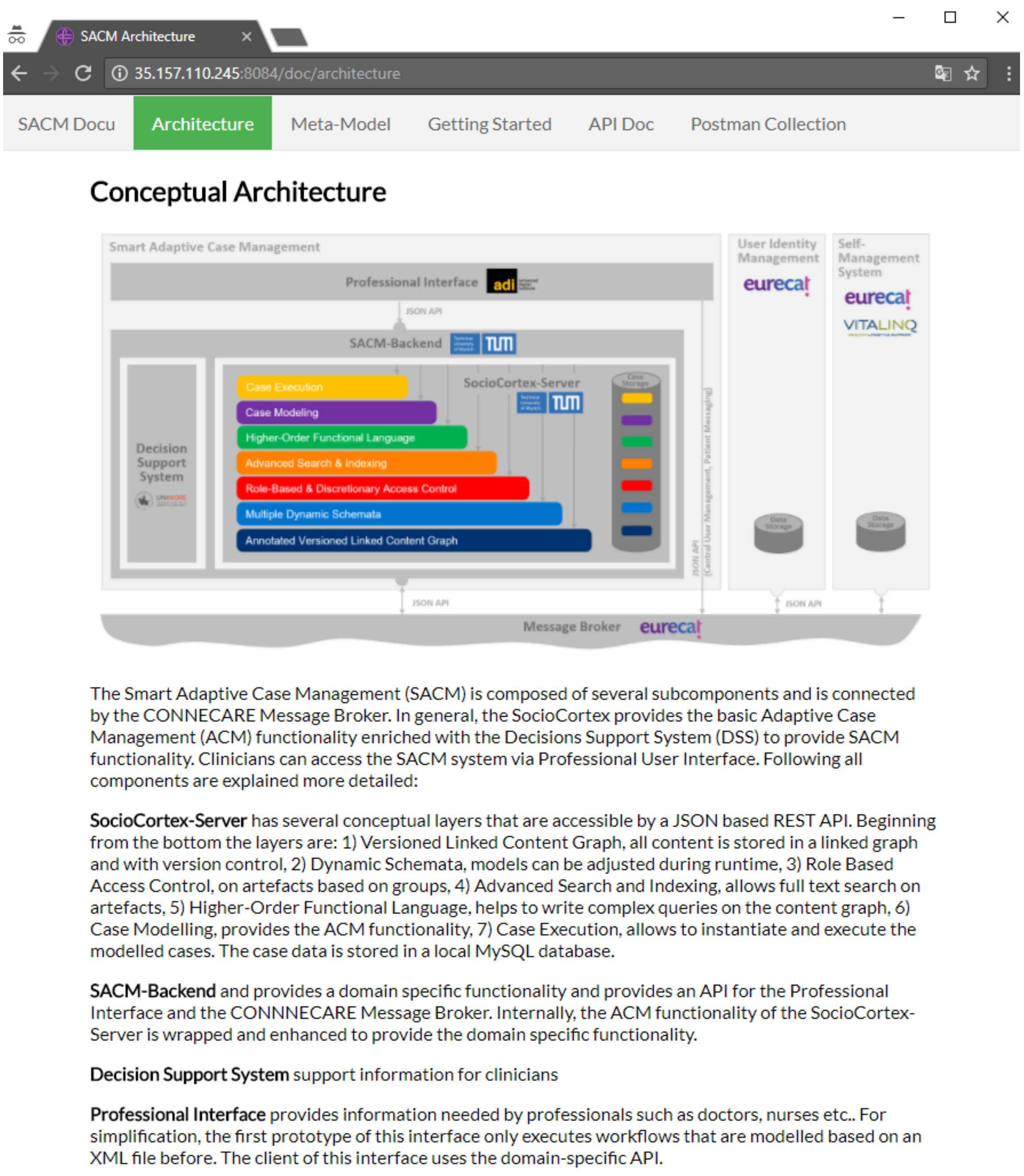
1. Initialize testcase

POST<domain>::<port>/api/import

To simplify the testing, this route creates a case definition and instantiates the case (this request may take long). Be aware, the imported case is incrementally changed according to the current implementation. The imported case definition is defined based on an XML schema. To customize this import please consider a local installation and modify the XML file accordingly.

Figure 20: *SACM documentation landing page navigation menu.*

[3] http://test.connecare.eu:8084/doc



Conceptual Architecture

The Smart Adaptive Case Management (SACM) is composed of several subcomponents and is connected by the CONNECARE Message Broker. In general, the SocioCortex provides the basic Adaptive Case Management (ACM) functionality enriched with the Decisions Support System (DSS) to provide SACM functionality. Clinicians can access the SACM system via Professional User Interface. Following all components are explained more detailed:

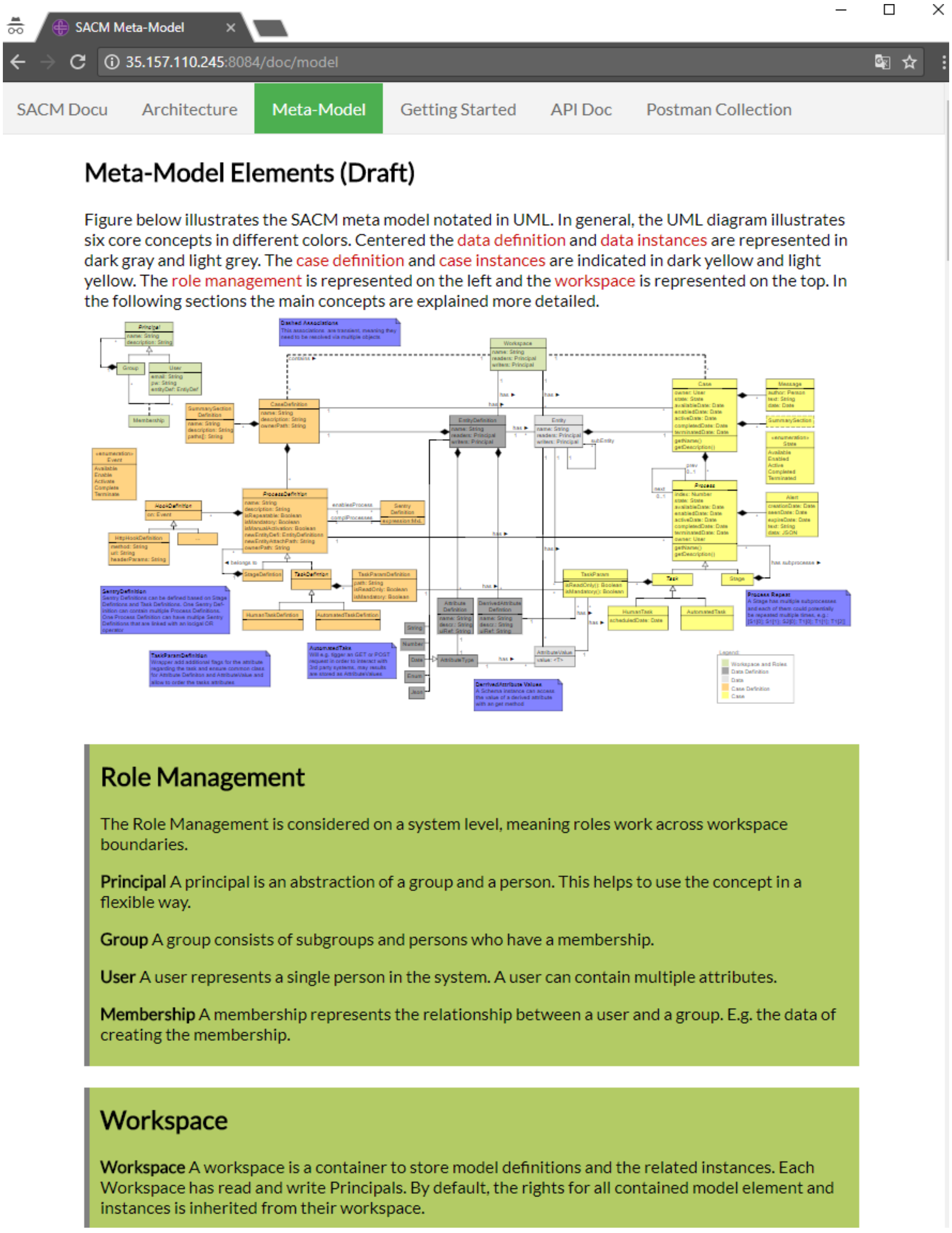
SocioCortex-Server has several conceptual layers that are accessible by a JSON based REST API. Beginning from the bottom the layers are: 1) Versioned Linked Content Graph, all content is stored in a linked graph and with version control, 2) Dynamic Schemata, models can be adjusted during runtime, 3) Role Based Access Control, on artefacts based on groups, 4) Advanced Search and Indexing, allows full text search on artefacts, 5) Higher-Order Functional Language, helps to write complex queries on the content graph, 6) Case Modelling, provides the ACM functionality, 7) Case Execution, allows to instantiate and execute the modelled cases. The case data is stored in a local MySQL database.

SACM-Backend and provides a domain specific functionality and provides an API for the Professional Interface and the CONNECARE Message Broker. Internally, the ACM functionality of the SocioCortex-Server is wrapped and enhanced to provide the domain specific functionality.

Decision Support System support information for clinicians

Professional Interface provides information needed by professionals such as doctors, nurses etc.. For simplification, the first prototype of this interface only executes workflows that are modelled based on an XML file before. The client of this interface uses the domain-specific API.

Figure 21: *Smart Adaptive Case Management Architecture overview that includes a high-level description of each subcomponent.*



Meta-Model Elements (Draft)

Figure below illustrates the SACM meta model notated in UML. In general, the UML diagram illustrates six core concepts in different colors. Centered the data definition and data instances are represented in dark gray and light grey. The case definition and case instances are indicated in dark yellow and light yellow. The role management is represented on the left and the workspace is represented on the top. In the following sections the main concepts are explained more detailed.

Role Management

The Role Management is considered on a system level, meaning roles work across workspace boundaries.

Principal A principal is an abstraction of a group and a person. This helps to use the concept in a flexible way.

Group A group consists of subgroups and persons who have a membership.

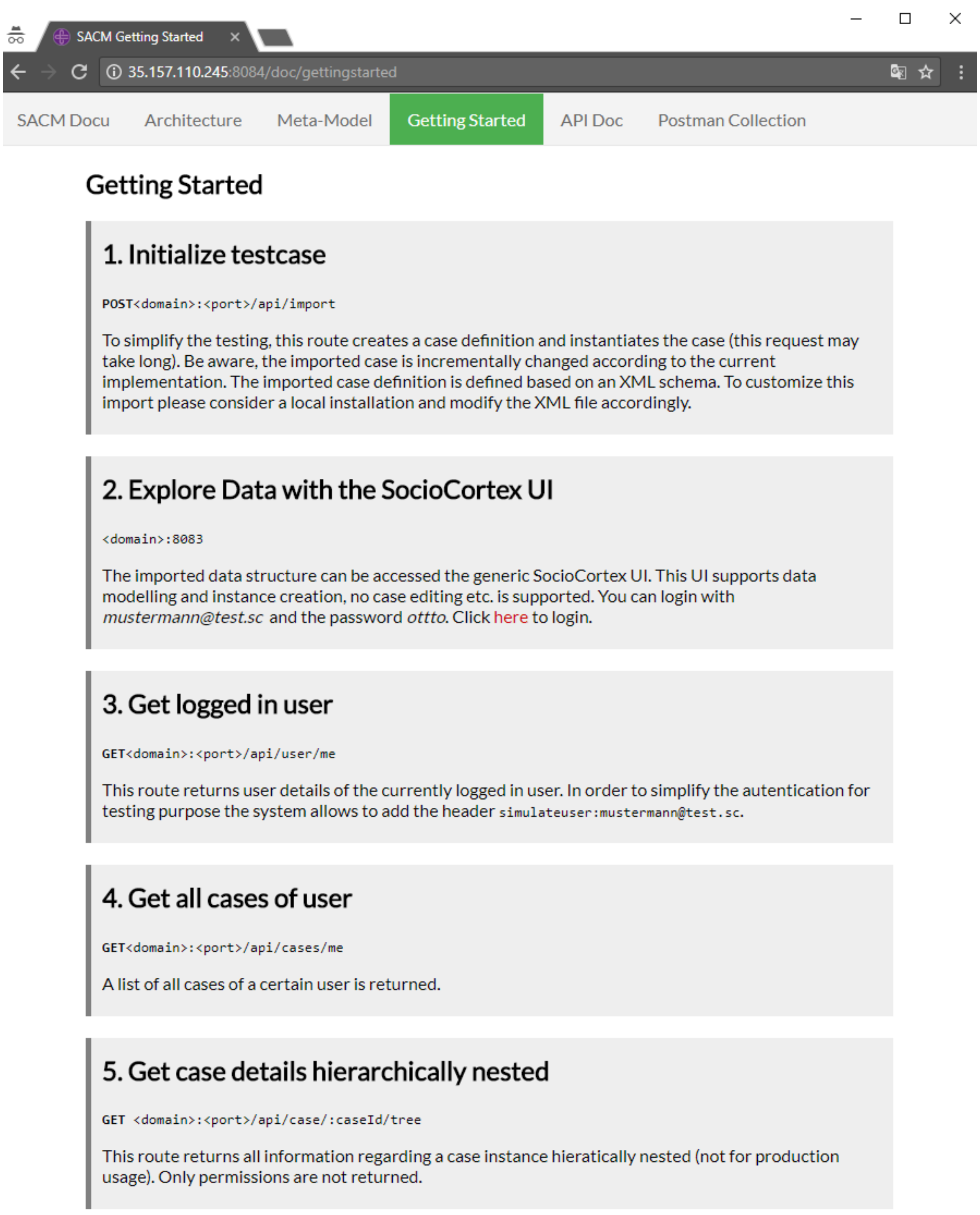
User A user represents a single person in the system. A user can contain multiple attributes.

Membership A membership represents the relationship between a user and a group. E.g. the data of creating the membership.

Workspace

Workspace A workspace is a container to store model definitions and the related instances. Each Workspace has read and write Principals. By default, the rights for all contained model element and instances is inherited from their workspace.

Figure 22: *Meta-Model documentation page shows an UML diagram that illustrate the conceptual classes. The different conceptual parts are indicated with different colours. A short description in the related colour explains each conceptual element more detailed. At the bottom of this page (not visible) a sample XML case definition files is illustrated.*



Getting Started

1. Initialize testcase

```
POST<domain>:<port>/api/import
```

To simplify the testing, this route creates a case definition and instantiates the case (this request may take long). Be aware, the imported case is incrementally changed according to the current implementation. The imported case definition is defined based on an XML schema. To customize this import please consider a local installation and modify the XML file accordingly.

2. Explore Data with the SocioCortex UI

```
<domain>:8083
```

The imported data structure can be accessed the generic SocioCortex UI. This UI supports data modelling and instance creation, no case editing etc. is supported. You can login with *mustermann@test.sc* and the password *ottto*. Click here to login.

3. Get logged in user

```
GET<domain>:<port>/api/user/me
```

This route returns user details of the currently logged in user. In order to simplify the autentication for testing purpose the system allows to add the header `simulateuser:mustermann@test.sc`.

4. Get all cases of user

```
GET<domain>:<port>/api/cases/me
```

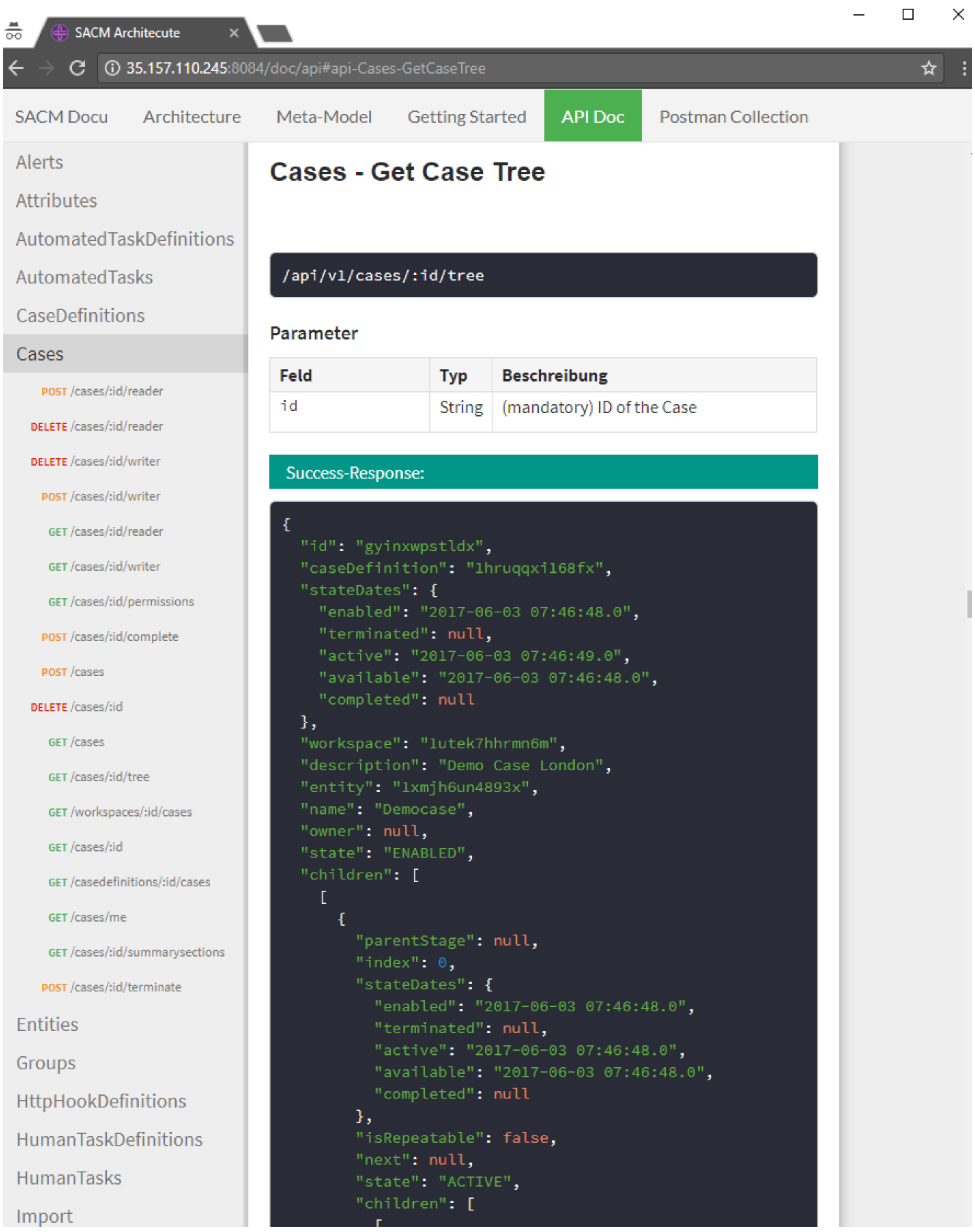
A list of all cases of a certain user is returned.

5. Get case details hierarchically nested

```
GET <domain>:<port>/api/case/:caseId/tree
```

This route returns all information regarding a case instance hieratically nested (not for production usage). Only permissions are not returned.

Figure 23: *Getting started tutorial – explains how to start developing with the SACM system.*



Cases - Get Case Tree

```
/api/v1/cases/:id/tree
```

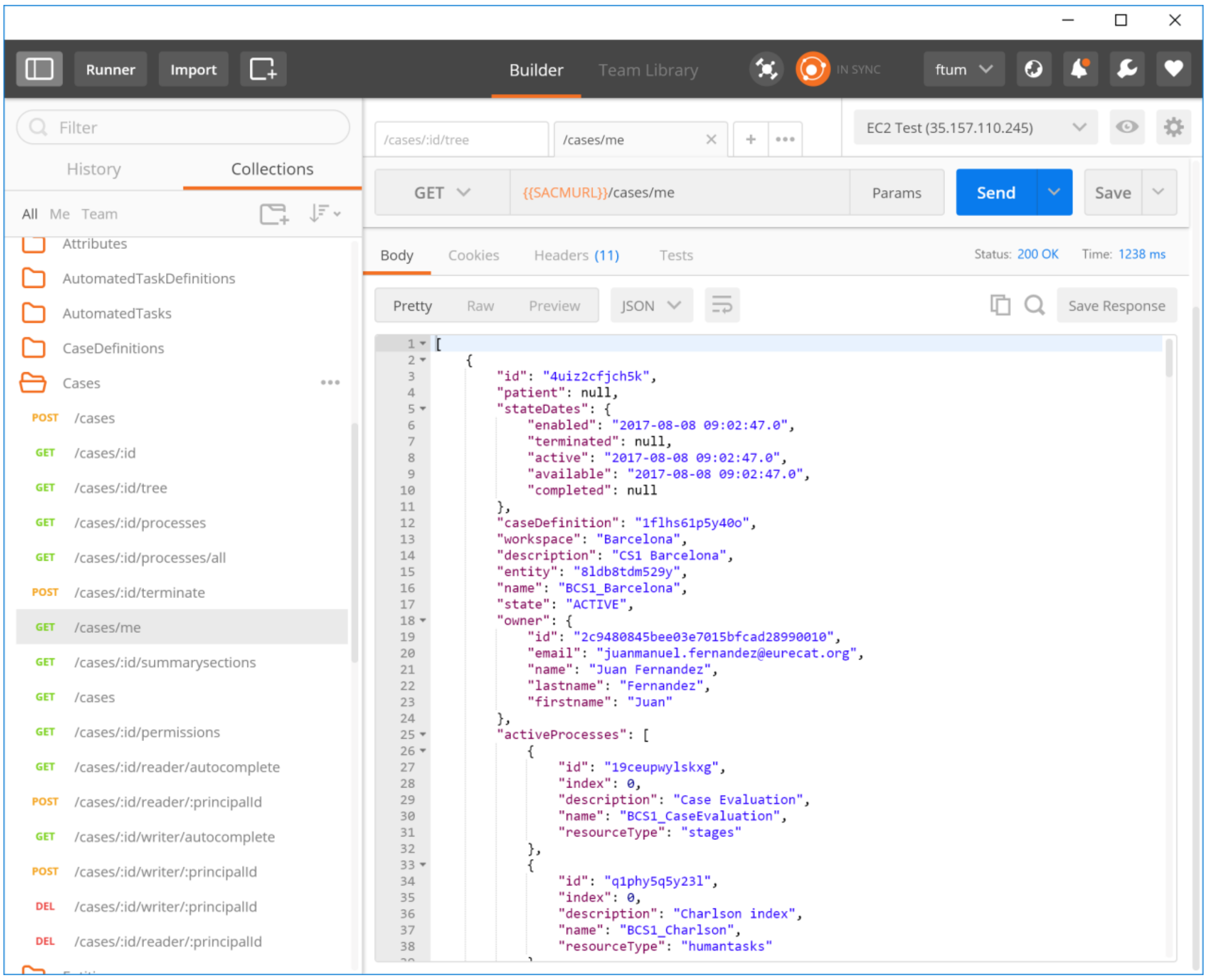
Parameter

Feld	Typ	Beschreibung
id	String	(mandatory) ID of the Case

Success-Response:

```
{
  "id": "gyinxwpstldx",
  "caseDefinition": "1hruqqxil68fx",
  "stateDates": {
    "enabled": "2017-06-03 07:46:48.0",
    "terminated": null,
    "active": "2017-06-03 07:46:49.0",
    "available": "2017-06-03 07:46:48.0",
    "completed": null
  },
  "workspace": "1utek7hhrmn6m",
  "description": "Demo Case London",
  "entity": "1xmjh6un4893x",
  "name": "Democase",
  "owner": null,
  "state": "ENABLED",
  "children": [
    [
      {
        "parentStage": null,
        "index": 0,
        "stateDates": {
          "enabled": "2017-06-03 07:46:48.0",
          "terminated": null,
          "active": "2017-06-03 07:46:48.0",
          "available": "2017-06-03 07:46:48.0",
          "completed": null
        },
        "isRepeatable": false,
        "next": null,
        "state": "ACTIVE",
        "children": [
```

Figure 24: *API documentation generated based on annotated API endpoints. Sample requests and responses are illustrated including their parameters.*



Attributes
AutomatedTaskDefinitions
AutomatedTasks
CaseDefinitions
Cases

POST	/cases
GET	/cases/:id
GET	/cases/:id/tree
GET	/cases/:id/processes
GET	/cases/:id/processes/all
POST	/cases/:id/terminate
GET	/cases/me
GET	/cases/:id/summarysections
GET	/cases
GET	/cases/:id/permissions
GET	/cases/:id/reader/autocomplete
POST	/cases/:id/reader/:principalId
GET	/cases/:id/writer/autocomplete
POST	/cases/:id/writer/:principalId
DEL	/cases/:id/writer/:principalId
DEL	/cases/:id/reader/:principalId

```
[
    {
        "id": "4uiz2cfjch5k",
        "patient": null,
        "stateDates": {
            "enabled": "2017-08-08 09:02:47.0",
            "terminated": null,
            "active": "2017-08-08 09:02:47.0",
            "available": "2017-08-08 09:02:47.0",
            "completed": null
        },
        "caseDefinition": "1flhs61p5y40o",
        "workspace": "Barcelona",
        "description": "CS1 Barcelona",
        "entity": "8ldb8tdm529y",
        "name": "BCS1_Barcelona",
        "state": "ACTIVE",
        "owner": {
            "id": "2c9480845bee03e7015bfcad28990010",
            "email": "juanmanuel.fernandez@eurecat.org",
            "name": "Juan Fernandez",
            "lastname": "Fernandez",
            "firstname": "Juan"
        },
        "activeProcesses": [
            {
                "id": "19ceupwylskxg",
                "index": 0,
                "description": "Case Evaluation",
                "name": "BCS1_CaseEvaluation",
                "resourceType": "stages"
            },
            {
                "id": "q1phy5q5y23l",
                "index": 0,
                "description": "Charlson index",
                "name": "BCS1_Charlson",
                "resourceType": "humantasks"
```

Figure 25: *Postman[4] collection[5] of SACM routes that is online available for simplification of the development and testing.*

[4] https://www.getpostman.com/

[5] https://documenter.getpostman.com/view/1815062/sacm/6Z3tXf7

5. Glossary

ACM	Adaptive Case Management
API	Application Programming Interface
CMMN	Case Management Model and Notation
Docker	provides an additional layer of abstraction and automation of operating-system-level virtualization
DSS	Decision Support System
EC2	Amazon Elastic Cloud
GIT	is a distributed Version Control System
JWT	Json Web Token
MSB	Message Broker
Postman	is a HTTP Request composer
SACM	Smart Adaptive Case Management
SC	SocioCortex, successor of Darwin
XML	Extensible Markup Language

6. List of Figures

7. References

Matheus Hauder, R. K. (2015). Empowering End-Users to Collaboratively Structure. *18th International Conference on Business Information Systems (BIS)*.

Matthes, F. C. (2011). Hybrid Wikis: Empowering Users to Collaboratively Structure Information. *ICSOFT (1) 11*, S. 250-259.