



Hochschule Darmstadt  
Fachbereich Informatik

# **Integrating Clinical Decision Support in a Product Line for Electronic Health Record Management**

Abschlussarbeit zur Erlangung des akademischen Grades  
Master of Science (M. Sc.)

vorgelegt von

Daniel Ebanja (726045)

Referent: Prof. Dr. Bernhard Humm  
Korreferent: Prof. Dr. Ralf Hahn

Ausgabedatum: 15.02.2018  
Abgabedatum: 15.08.2018

# Erklärung

Ich versichere hiermit, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die im Literaturverzeichnis angegebenen Quellen benutzt habe. Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder noch nicht veröffentlichten Quellen entnommen sind, sind als solche kenntlich gemacht. Die Zeichnungen oder Abbildungen in dieser Arbeit sind von mir selbst erstellt worden oder mit einem entsprechenden Quellennachweis versehen. Diese Arbeit ist in gleicher oder ähnlicher Form noch bei keiner anderen Prüfungsbehörde eingereicht worden.

Darmstadt, 15. August 2018

---

# Abstract

Der 1999 erschienene Bericht "To Err is Human" des Institute of Medicine zeigte auf, dass medizinische Fehler zu mindestens 98.000 Todesfällen pro Jahr führen. Zur Lösung dieses Problems wurde die Anwendung von Informationstechnologie im Gesundheitswesen vorgeschlagen und weitgehend übernommen.

Elektronische Patientenaktensysteme (auch EHR-Systeme genannt) und computergestützte Auftragsannahmesysteme (auch CPOE-Systeme genannt) sind für die Verbesserung der Patientensicherheit und der Qualität der Gesundheitsversorgung unerlässlich. Auch die Einführung klinischer Entscheidungsunterstützungssysteme (auch CDS-Systeme genannt) zur Fehlervermeidung und -erkennung ist eine Hauptvoraussetzung für den sinnvollen Einsatz von EHR-Systemen.

Der verbreitete Einsatz von EHR- und CDS-Systemen in der Praxis führt dazu, dass Anwendungsanbieter gleichzeitig mit mehreren Anforderungen für ähnliche Produkte (EHR- und CDS-Systeme) konfrontiert werden.

Zur Lösung dieser Problematik schlägt die vorliegende Arbeit die Einführung eines Produktlinienansatzes vor, indem sie darauf abzielt, CDS-Systeme in (vorhandene) EHR-Anwendungen einer Produktlinie über Core Assets zu integrieren.

Die Literatur zeigt, dass der Informationsbedarf der Ärzte am Behandlungsort mit der Diagnose von Krankheiten, der Suche nach den besten Therapien und Medikamentenverordnungen zusammenhängt. Ärzte decken ihren Informationsbedarf dabei hauptsächlich mithilfe internetbasierter Ressourcen.

Hinsichtlich der Architektur von CDS-Systemen zeigen Studien, dass die IT-Branche im Gesundheitswesen weitgehend SOA-basierte Ansätze anwendet. Diese Thèse setzt deshalb auf SOA-basierte Ansätze.

In dieser Thèse werden CDS Core Assets entworfen und entwickelt, welche patientenspezifische Empfehlungen durch den Einsatz von REST-Anwendungsprogrammierschnittstellen (APIs) von medizinischen Informationsabrufressourcen wie PubMed, RxNav, OpenFDA usw. liefern.

Die CDS Core Assets stellen Funktionen zur Überprüfung von Wechselwirkungen mit anderen Arzneimitteln und unerwünschten Arzneimittelwirkungen zur Verfügung. Sie bieten auch Funktionen für die Suche nach klinischen Studien, evidenzbasierte medizinische Leitlinien und medizinische Publikationen. Die beispielhafte Umsetzung der CDS Core Assets wird anhand einer bestehenden Beispiel-EHR-Anwendung mit integriertem CDS demonstriert.

# Abstract

The 1999 report of the institute of medicine “To Err is Human” revealed, that medical errors account for at least 98000 deaths per year. To address this problem, the application of information technology to the field of health care has been proposed and widely adopted. Electronic Health Record (EHR) systems and Computerized Physician Order Entry (CPOE) systems are essential for improving patient safety and the quality of health care. Also, the adoption of clinical decision support systems (CDS-Systems) for error prevention and detection is a main requirement for meaningful use of EHR systems. The wide adoption of EHR systems and CDS-Systems result in application providers being faced with demands for several similar products at the same time.

This thesis aims to addresses the problems faced by physicians and application providers.

Literature reveals, that physicians’ needs at the point of care are related to diagnosing patients’ issues, finding best therapies and drug prescriptions and that Physicians refer mostly to internet-based resources to meet their information needs.

Furthermore, literature is reviewed to identify architectural approaches for developing CDS-Systems. An evaluation of found approaches reveal that SOA-based approaches best suit the requirements of this thesis.

This thesis therefore proposes an approach for integrating CDS into EHR via SOA-based CDS-Services. The proposed concept places an API in front of the CDS-Services, making them available via a web service. In this way, the CDS-Services are accessible for EHR applications and other health information systems e.g. CPOE. The CDS-Services include a drug interaction service, a drug adverse events service, a clinical trials service, an EBM guideline service and a literature service. The CDS-Services provide patient-specific recommendations by accessing other medical information services, like PubMed, RxNav, OpenFDA, ClinicalTrials.gov etc.

The proposed concept is applied to provide CDS-Services in EHR applications of the SAGE-CARE product line. To this purpose, core asset development activities are performed based on the SPLE Framework of (Pohl et al. 2005). The feasibility of the proposed concept and the developed CDS core assets is asserted by prototyping a sample EHR application with integrated CDS-Services.

## Contents

1	Introduction .....	1
2	Problem statement .....	3
3	Background .....	4
3.1	Electronic Health Records .....	4
3.2	Health Level Seven Reference Information Model .....	4
3.3	Clinical Decision Support Systems .....	5
3.4	Software product lines .....	8
3.4.1	Software product line engineering .....	9
3.4.2	Domain engineering .....	10
3.4.3	Application engineering .....	14
3.5	Cloud computing and Multitenancy .....	15
4	CDS Information .....	17
4.1	Physicians' information needs .....	17
4.2	Information retrieval resources for CDS content .....	19
5	Architectures for Clinical Decision Support Systems .....	21
5.1	Evolution of CDS architectures .....	21
5.1.1	Stand-alone CDS-Systems .....	22
5.1.2	Integrated CDS-Systems .....	22
5.1.3	Standards-based CDS-Systems .....	22
5.1.4	Service oriented CDS-Systems .....	23
5.2	CDS-System design components .....	24
6	SOA-based CDS-Services .....	26
6.1	Drug interaction service .....	28
6.2	Drug adverse events service .....	29
6.3	Clinical trial service .....	30
6.4	Literature service .....	31
6.5	Terminology service .....	33

7	The SAGE-CARE product line use case.....	36
7.1	Product Management.....	36
7.2	Domain requirements engineering.....	36
7.2.1	Commonality analysis .....	36
7.2.2	Variability analysis .....	38
7.3	Domain design.....	41
7.4	Domain realization .....	45
7.4.1	SAGE-CARE_Core project.....	46
7.4.2	SAGE-CARE_RESTful_WebAPI.....	47
7.4.3	SAGE-CARE_Client project.....	47
7.5	Domain Testing using Sample Application Strategy .....	55
7.5.1	Overview of the sample EHR application .....	55
7.5.2	Core project .....	56
7.5.3	RESTful_WebAPI.....	57
7.5.4	Client project .....	59
8	Evaluation.....	61
9	Related Work.....	64
10	Conclusion and Future Work .....	65
10.1	Conclusion.....	65
10.2	Future Work.....	66
11	Appendices .....	67
11.1	Appendix A.....	67
11.2	Appendix B.....	69
11.3	Appendix C.....	70
11.4	Appendix D.....	71
11.5	Appendix E .....	72
11.6	Appendix F .....	73
11.7	Appendix H.....	74
11.8	Appendix I.....	75

11.9	Appendix J.....	76
12	Publication bibliography .....	77

# List of Figures

Figure 1 Essential Product Line Activities (Clements and Northrop 2009, p. 30).....	9
Figure 2. The software product line engineering framework (Pohl et al. 2005, p. 22) .....	10
Figure 3 Graphical notation for variability models (Pohl et al. 2005, p. 85) .....	12
Figure 4 Authentication mechanism variation point using OVM vs UML .....	13
Figure 5. What is SaaS (Adapted from (What is SaaS? Software as a Service   Microsoft Azure)) .....	15
Figure 6. Multitenant App with database per tenant .....	16
Figure 7. Multitenant App with multi-tenant database.....	16
Figure 8 A schematic drawing of the four-phase model for clinical decision support (Wright and Sittig 2008) .....	21
Figure 9 Sample CDS system architecture (Adapted from (Rodriguez-Loya and Kawamoto 2016)).....	23
Figure 10 A conceptual model of CDS design components and their interactions with the host application environment. (Greenes 2014b) .....	24
Figure 11 CDS-Services in a SOA-based Health information systems environment .....	26
Figure 12 CDS-Services architecture .....	27
Figure 13. OVM enabled application services .....	38
Figure 14 OVM perceived locus of control.....	39
Figure 15 OVM supported medical specialties .....	39
Figure 16 OVM enabled CDS-Services .....	40
Figure 17 OVM drug interaction information sources .....	40
Figure 18 SAGE-CARE SPL Reference Architecture with CDS-Services .....	41
Figure 19 CDS core assets in SAGE-CARE Presentation Layer .....	44
Figure 20 SAGE-CARE SPL projects.....	45
Figure 21 CDS Core Assets in SAGE-CARE_Core project .....	46
Figure 22 CDS Core Assets in SAGE-CARE_Client project .....	48
Figure 23 CDS-Services user interaction interface .....	49
Figure 24 Drug interaction service user interface .....	50
Figure 25 Additional information in drug interaction service user interface .....	50
Figure 26 Autocomplete service in drug interaction user interface.....	51
Figure 27 Entering Medications for interaction check in drug interaction service user interface .....	51
Figure 28 Autosuggest service in drug adverse event service user interface .....	52
Figure 29 Verifying drug adverse events of non-prescribed drugs .....	52



Figure 30 Clinical trial service user interface.....	52
Figure 31 ClinicalTrials.gov patient-specific result page.....	53
Figure 32 EBM Guidelines service user interface .....	53
Figure 33 Literature service user interface .....	54
Figure 34 Terminology service autosuggest in EHR medication field .....	55
Figure 35 Breast cancer SampleIssue .....	56
Figure 36 Introducing the Sample folder for Sample application classes .....	56
Figure 37 CDS-Service classes for Sample application in SAGE-CARE core project.....	57
Figure 38 Creating the Sample application controller .....	58
Figure 39 Sample application core assets in the client project.....	59
Figure 40 Drug interaction service sequence diagram .....	72
Figure 41 Drug adverse events service sequence diagram .....	73
Figure 42 SAGE-CARE Information Model.....	74
Figure 43 HL7 Reference Information Model (HL7 RIM) .....	76

# List of Tables

Table 1 Principal purposes for CDS, and the key methodologies used (Greenes 2014b) .....	7
Table 2. List of information needs of physicians and nurses commonly mentioned in the review of literature (Clarke et al. 2013) .....	17
Table 3 CDS-System information services and addressed information needs.....	18
Table 4. List of information sources utilized by physician and nurses (Clarke et al. 2013) ....	19
Table 5 CDS-Services, selected information retrieval resources .....	20
Table 6 Advantages and disadvantages of Stand-alone CDS-Systems (Wright and Sittig 2008) .....	22
Table 7 Advantages and disadvantages of tightly coupled integrated CDS-Systems (Wright and Sittig 2008) .....	22
Table 8 Advantages and disadvantages of standards-based CDS-Systems (Wright and Sittig 2008).....	22
Table 9 CDS-Services interface specifications .....	28
Table 10 Principal purposes for CDS, and the key methodologies used (Greenes 2014a) .....	37
Table 11 Overview of available interface implementations for CDS-Services.....	47
Table 12 Drug Information Data Sources (Johannes Idelhauser 2016).....	68
Table 13 Identified data sources for the clinical trial locator (Johannes Idelhauser 2016) .....	69
Table 14 EBM Recommendation Sources (Johannes Idelhauser 2016).....	70
Table 15 Literature Service Data Sources (Johannes Idelhauser 2016) .....	71

# Listings

Listing 1 Using the drug interaction service.....	28
Listing 2 Drug interaction service interface - IDrugInteractionService .....	29
Listing 3 Drug interaction service interface response data format – DrugInteraction .....	29
Listing 4 Drug adverse events service interface – IDrugAdverseEventsService .....	29
Listing 5 Drug adverse events service response data format - DrugAdverseEvents.....	29
Listing 6 Drug adverse events service response data format – AdverseEvent.....	30
Listing 7 Clinical trial service interface - IClinicalTrialService .....	30
Listing 8 Clinical trial service parameter data format – ClinicalTrialQuery.....	30
Listing 9 Specification of the gender data type .....	31
Listing 10 Literature service interface – ILiteratureService.....	31
Listing 11 EHR mapping for literature service – SearchField .....	31
Listing 12 EhrField data type for EHR mapping in literature service .....	32
Listing 13 Literature service response data format - Literature .....	32
Listing 14 Terminology service interface - ITerminologyService .....	33
Listing 15 Terminology service response data format -Term .....	34
Listing 16 Terminology service parameter data format .....	35
Listing 17 Using the drug interaction service.....	47
Listing 18 One liner for displaying CDS-Services in EHR application .....	49
Listing 19 Using the Sample drug interaction CDS-Service - SampleDrugInteractionService .....	57
Listing 20 Specifying the Sample application Web API in SampleCDSServicesController.cs .....	58
Listing 21 SamplePatientRecordView.html - introducing CDS-Services UIs in the sample EHR application .....	60
Listing 22 SAGE-CARE_Client configuration file template CDS-Systemervices.html .....	75

# Acronyms

ANSI	American National Standard Institute
API	Application Programming Interface
CDS-Service	Clinical Decision Support Service
CDS	Clinical Decision Support
CDS-System	Clinical Decision Support System
CIS	Clinical Information System
CPOE	Computer-based Physician Order Entry
CRUD	Create, Red, Update, Delete
DPL	Dynamic product line
EBM	Evidence-Based Medicine
EHR	Electronic Health Record
HL7	Health Level 7
ISO	International Organization for Standardization
IaaS	Infrastructure as a Service
IIS	Internet Information Services
Ids	Identifiers
JSON	JavaScript Object Notation
LOC	Lines of Code
MIS	Medical Information Service
MS	Medical Specialty
NCCN	National Comprehensive Cancer Network
NCCN	National Comprehensive Cancer Network
NLM	U.S. National Library of Medicine
OVM	Orthogonal variability model
ORM	Object-relational mapper
PaaS	Platform as a Service
PL	Product Line
REST	Representational State Transfer
RIM	Reference Information Model
RSA	Rivest-Shamir-Adleman
SaaS	Software as a Service
SOA	Service-Oriented Architecture
SQL	Structured Query Language
SAGE-CARE	SemAntically integrating Genomics with Electronic health records for Cancer CARE
SAS	Sample Application Strategy
SPL	Software Product Line
SPLE	Software Product Line Engineering
Time to market	TTM
TS	Technical Service
UASD	University of Applied Sciences Darmstadt
UML	Unified modeling language
VM	Virtual MachineVirtual Machine

# 1 Introduction

The 1999 report of the institute of medicine “To Err is Human”, revealed that medical errors account for at least 98000 deaths per year (Kohn et al. 2000). Health informatics focuses on the application of information technology to the field of Health care with the goal of improving patient safety and the quality of care. (HSRIC: Health Informatics 2018). Electronic Health Record (EHR) systems, Computer-based Physician Order Entry (CPOE) systems and Clinical Decision Support Systems (CDS-System) are suggested approaches to achieve improvements in patient safety and the quality of health care (Bates et al. 1998; Kohn et al. 2000; National Academies Press (US) 2001).

An EHR can be defined as “a digital version of a patient’s paper chart.” (HealthIT.gov 2018) EHRs “contain a patient’s medical history, diagnoses, medications ” (HealthIT.gov 2018). EHRs promote the implementation of clinical decision support to assist physicians at the point of care. A CDS-System provides recommendations to assist physicians in various aspects of healthcare that include but are not limited to: diagnosing health conditions, finding best therapies and drug prescriptions etc.

In 2014, the SAGE-CARE (SemAntically integrating Genomics with Electronic health records for Cancer CARE) project was launched. SAGE-CARE is funded by the European Union’s Horizon 2020 research and innovation program under the Marie Skłodowska-Curie grant agreement No 644186. The aim of the project is “to bring together subject matter experts from the academic and non-academic sectors to create a holistic informatics platform for rapidly integrating genomic sequences, electronic health records (EHRs) and research repositories to enable personalised medicine strategies for malignant melanoma treatment.” (COMM/RTD)

In partnership with NSilico Lifescience Ltd. Within the SAGE-CARE Project, work has been performed at the University of Applied Sciences in Darmstadt (UASD) to develop an EHR with an integrated CDS-System for malignant melanoma. The EHR and CDS-System use a data model conform to the Health Level 7 (HL7) Reference Information Model (RIM).

This thesis emerges as part of the efforts performed at the UASD. Several thesis and papers have been written to present the progress of the work at this site. (Humm and Walsh 2015) present a simplified EHR application for management of melanoma patient data. In (Beez et al. 2015) present a semantic automatic suggest service that enhances data entry into the free text fields of the EHR and enforces the use of a normalized medical vocabulary. In (Ulrich Beez 2015), the EHR is extended, to integrate a CDS-System that retrieves relevant medical publications based on patient-specific data from the EHR. (Johannes Idelhauser 2016),

discusses the information needs of physicians at the point of care, and physicians' information seeking behaviour. (Johannes Idelhauser 2016) extends the CDS-System functionality of the EHR by implementing a drug interaction service. (Tino Landmann 2017) presents a medical information service that displays an evidence-based medical guideline for the treatment of malignant melanoma.

Furthermore, (Patrick Spitzer 2016) presents concepts for a dynamic product line (DPL) of EHR applications for cancer care. The presented product line approach enables parallel development of multiple EHR applications targeting differing medical specialities. Advantages of adopting a product line are e.g. product quality enhancements, improved time to market (TTM), reduction of development costs and maintenance efforts (Pohl et al. 2005).

The aim of this thesis is to integrate CDS into EHR applications of a product line via reusable clinical decision support (CDS) services. The CDS-Services on the one hand, aim to meet the needs of physicians at the point of care, and, on the other hand, those of the product line owner via reusability. Reusability promotes the CDS-Services to product line core assets and provides for improved TTM, reduction of development costs and maintenance efforts.

To achieve the goals of this thesis, the following questions are posed and addressed:

- What are the information needs of physicians at the point of care?
- What information behaviour do physicians adopt to meet their information needs?
- Which internet-based information retrieval resources provide CDS information and how can they be used to meet physicians' information needs.
- What approaches can be adopted for developing CDS? How do these approaches support flexibility and reusability?

The remaining chapters of this thesis are structured as follows: Chapter 2, specifies the problem statement and presents requirements that are aimed to be met. Chapter 3 provides relevant fundamentals, In Chapter 4, physicians' information needs and their information seeking behaviour are identified. Furthermore, some CDS-Services are proposed to address physicians' information needs. Chapter 5 presents and evaluates approaches for developing CDS-Systems. Upon identifying a suitable architecture, chapter 6 continues to discuss the proposed concept for system independent CDS-Services. The proposed concept is evaluated in Chapter 7, that discusses the development of CDS-Services and their integration into a Sample application. Chapter 8 evaluates the proposed concept based on the thesis requirements and CDS implementation guidelines from the guidelines project (BVBA 2017b). Related Work is mentioned in Chapter 9. Chapter 10 concludes this thesis and mentions considerations for future work.

## 2 Problem statement

The goal of this thesis is to integrate CDS into EHR applications via reusable CDS-Services. The CDS-Services aim to meet the information needs of physicians (from different medical specialties) at the point of care, and that of the product line owner. The following fundamental requirements were identified:

The CDS-Services shall:

1. Be integrated into the clinical workflow, i.e. it shall operate unobtrusively to avoid alert-fatigue.
2. Provide relevant information.
3. Provide latest information.
4. Pro-actively search and provide decision support without requiring entry of already existing patient data.
5. Be intuitive and easy to use.
6. Be comprehensive.
7. Have low response times for interactions with any of the provided CDS-Services.

The architecture of the product line shall:

8. Support easy integration of CDS-Services into EHR applications.
9. Enable introducing new CDS-Services, with moderate implementation effort.
10. Enable introducing new information retrieval resources, with moderate implementation effort.

## **3 Background**

### **3.1 Electronic Health Records**

Health care is a collective effort that involves various stakeholders and clinicians from different domains and organizations. EHRs aim to enhance health care and therefore provide centralized patient information, facilitating patient-information sharing between health care providers and organizations, e.g. laboratories, pharmacies etc. (HealthIT.gov 2018).

An EHR is defined as “a digital version of a patient’s paper chart.” (HealthIT.gov 2018) EHRs “contain a patient’s medical history, diagnoses, medications, treatment plans, immunization dates, allergies, radiology images, and laboratory and test results” (HealthIT.gov 2018). The International Organization for Standardization (ISO) defines an EHR as “a repository of information regarding the health status of a subject of care, in computer processable form, stored and transmitted securely and accessible by multiple authorized users, having a standardized or commonly agreed logical information model that is independent of EHR systems and whose primary purpose is the support of continuing, efficient and quality integrated health care” (ISO/TR 20514:2005).

This thesis adopts both afore-mentioned definitions of an EHR (digital patient data). The terms EHR application and EHR system are used synonymously to refer to the software used to manage EHRs.

Benefits of EHRs, when properly used, include but are not limited to, “improved Patient Care, Increase Patient Participation, Improved Care Coordination, Improved Diagnostics & Patient Outcomes, Practice Efficiencies and Cost Savings” (HealthIT.gov 2018). EHRs also promote the implementation of clinical decision support to assist physicians at the point of care.

### **3.2 Health Level Seven Reference Information Model**

The ISO definition of an EHR emphasizes the need for a standardized or commonly agreed logical information model. This model serves as a vehicle for information sharing across the different stakeholders involved in health care.

The HL7 RIM is an ANSI (American National Standard Institute) approved standard that provides an object model for management and sharing of EHR information (HL7 Standards; HL7 RIM).

The HL7 RIM is depicted in Figure 43. 6 classes form the core of the HL7 RIM namely:



- Entity: represents real world objects that participate in health care e.g. persons, organizations, devices, materials, places etc.
- Role: represents the function or position assumed by entities as they participate in health care e.g. doctor, nurse, patient etc.
- Act: represents activities, actions, or processes performed during health care.
- Participation: associates an Act to a Role, e.g. a consultant to a performed operation.
- RoleLink: represents a relationship between two separate roles, e.g. a patient and a consultant.
- ActRelationship: represents a relationship between two acts e.g. a laboratory test request and laboratory test result.

Although the use of RIM is advised, developers based on their needs, most likely use simplified or modified versions in EHR implementations (Greenes 2014a, p. 132). The EHR discussed in this thesis uses a modified version of the HL7 RIM as well. A level of abstraction is introduced including 2 classes, namely HL7Relationship (as superclass for all relationship classes) and HL7Object (as super class for Entity, Role and Act). Figure 42 depicts the information model used in this thesis. Despite the modifications, the information model remains conform to the HL7 RIM.

### **3.3 Clinical Decision Support Systems**

EHRs facilitate the use of patient data to assist physicians, by preventing errors during the administration of health care.

Clinical Decision Support (CDS) is defined as “the use of information and communication technologies (EHRs inclusive) to bring relevant knowledge to bear on the health care and well-being of a patient.” (Greenes 2014a, p. 8)

In this thesis, Clinical Decision Support Systems (CDS-Systems) are defined as software applications that assist physicians at the point of care (decision making time point) by providing patient-specific recommendations (Greenes 2014a, p. 8; Berner 2016b, p. 1).

A CDS-System provides recommendations to assist physicians in various aspects of health care that include but are not limited to: diagnosing health conditions, finding best therapies and drug prescriptions. In this thesis a CDS-System is viewed as an assembly of CDS-Services. CDS-Services provide recommendations to address health care aspects. A CDS-System is thus said to provide various CDS-Services e.g. a drug interaction service, drug adverse event service, clinical trial locator service, medical publication search service etc.

## Types of CDS-System

There are several schemes for classifying CDS-Systems in Literature. This thesis considers the following classifications.

### Intended user

Though traditionally intended for physicians, there has been a trend in the last decades for CDS-Systems targeting patients and other stakeholders (Col and Correa-de-Araujo 2014).

Depending on the targeted users (i.e. health care providers or consumers), the type of information and the mode of interaction with the CDS-System may vary considerably.

This thesis considers CDS for health care providers, particularly physicians.

### Purpose

CDS-Systems are also described in terms of pursued goals.

Purpose	Key Methodologies
Answering questions	Direct hyperlinks from context-specific settings, context-specific information retrieval, use of agents and information brokers, infobuttons as instance of the latter, or ultimately, a “personal guidance system”
Marking decisions	Gathering data, analyzing the data, and providing recommendations for assessments or actions
• Diagnosis	Bayes’ theorem, algorithmic computation, heuristic reasoning, statistical data mining/pattern recognition methods
• Test selection	Decision analysis, logical rules/appropriateness criteria, and logistic models and belief networks for risk prediction (e.g. for screening decisions)
• Choice of treatment	For choosing among alternatives, decision analysis, and logical rules/appropriateness criteria, including increasingly genotype considerations. For dose modifications for age or factors such as renal function, algorithmic computation. For dosimetry or dose distribution, algorithmic computation based on geometric and pharmacokinetic models, with use of heuristics and statistical methods for optimization
• Prognosis	Logistic regression, Markov modeling, survival analysis models, and quality of life assessment scoring methods
Optimizing process flow and workflow	Multistep algorithms, guidelines, and protocols, coordination of participants by workflow modeling, scheduling, and communication methods
Monitoring actions	Use of ECA <sup>1</sup> rules, with background detection of events, in real-time or asynchronously, logical evaluation of conditions, and issuing of messages. Events can be a user activity such as choice selection or data entry, a result arrival, or the passage of time
Focusing attention and enhancing visualization	Organization and presentation of items in data entry, display, or reporting applications. May be done by use of sequences to encourage intended behaviors, by a process flow model such as

---

<sup>1</sup> Event-condition-action rules

	an underlying guideline, and/or by visual groupings based on shared attributes such as purpose, medical subdomain, or application context. May also include dashboards, trend plots, or other summarization and visualization methods that make it easier to identify key elements needed for decision making
--	---

Table 1 Principal purposes for CDS, and the key methodologies used (Greenes 2014b)

The CDS-System discussed in this thesis pursues the purposes of answering questions, assist decision making by providing patient-specific recommendations, assist physicians in the choice of treatment, and finally monitoring actions.

### **Time at which CDS is provided**

CDS-System implementations also vary based on the time at which they provide decision support. This could either be “before, during, or after the clinical decision is made” (Berner 2016a, p. 2).

### **Interactivity**

Interactivity deals with whether the CDS-System is actively or passively provided. For example, a CDS-System can actively provide reminders and alerts that demand action from the physician or passively wait for the physician to require the information. In this thesis, the adopted approach is to passively provide CDS. The aim is to avoid acting obtrusively to the clinical workflow of the physician (see (Berner 2016a, p. 71)).

### **Degree of integration with EHR**

CDS-Systems can either be developed as standalone systems or be integrated into EHR applications. Both approaches have pros and cons, however, literature reveals that there has been a movement from standalone system architectures to service oriented architectures (see (Rodriguez-Loya and Kawamoto 2016)). This thesis adopts a service-oriented architectural approach for developing CDS-Systems, which is further discussed in Chapter 5.

### **Knowledge-Based or Nonknowledge-Based**

This is an architectural trait of CDS-Systems. A CDS-System might either encompass a knowledge base or not. A Knowledge base refers to a form of “compiled information that is often but not always, in the form of if-then rules.” (Berner 2016a, p. 3)

CDS-Systems without a knowledge base mainly rely on artificial intelligence that leverage machine learning techniques to provide CDS. Such CDS-Systems have not yet been widely

adopted due to the lack of transparency in applied reasoning steps (Holst et al. 2000). Physicians must understand how any given conclusions are met.

The CDS-System discussed in this thesis neither has a local knowledge base nor does it employ any local machine learning techniques to make recommendations. The CDS-System rather exploits the availability of web based medical resources. It consumes available REST APIs to retrieve CDS content and provide recommendations. The CDS-System adopts a service oriented architectural approach which is further discussed in Chapter 5.

### **3.4 Software product lines**

Product lines are ubiquitous in nowadays industry. For example, automotive and mobile phone manufacturers increasingly produce similar vehicles and mobile phones.

“A software product line is a set of software-intensive systems sharing a common, managed set of features that satisfy the specific needs of a particular market segment or mission and that are developed from a common set of core assets in a prescribed way”. (Clements and Northrop 2009, p. 5)

Core assets provide a common platform for product derivation and instantiation. Core assets are alternately referred to as, reusable artefacts (see (Pohl et al. 2005, p. 20)). The terms core assets and reusable artefacts are used synonymously in this thesis.

Core assets include all sorts of software development artefacts e.g. “the architecture, reusable software components, domain models, requirements statements, documentation and specifications, performance models, schedules, budgets, (test designs), test plans, test cases, work plans, and process descriptions.” (Pohl et al. 2005, p. 20; Clements and Northrop 2009, p. 14)

The benefits of adopting the software product line approach include:

- Production economies i.e. reduction of development costs
- Reduction of time-to-market
- Enhancement of product quality
- Mass customization support, resulting in
- Improved customer satisfaction etc. (Clements and Northrop 2009, pp. 17–27; Pohl et al. 2005, pp. 9–13)

### 3.4.1 Software product line engineering

Software product line engineering involves the systematic reuse of core assets for derivation of software products that are tailored to meet customers' needs. (Pohl et al. 2005, p. 14)

(Clements and Northrop 2009) identify three essential activities for SPLE are presented, namely Core asset development, Product development and Management activities.

Management activities are further subdivided into technical and organizational management. Organizational management provides resources to support SPLE e.g. competent personnel and funding. It also performs activities for Customer-Relationship-Management. Technical management on the other hand supervises core asset and product development activities. (Clements and Northrop 2009, p. 45)



Figure 1 Essential Product Line Activities (Clements and Northrop 2009, p. 30)

In this thesis, the SPLE Framework proposed by (Pohl et al. 2005) is used. It separates SPLE into two separate development processes namely, the Domain and Application engineering processes (Pohl et al. 2005, p. 20). These processes map the Core Asset and Product Development process proposed by (Clements and Northrop 2009). The SPLE Framework is depicted in Figure 2. The SPLE Framework does not specific a sequential order of execution for the processes and the processes are performed iteratively (Pohl et al. 2005, p. 23). The subsequent sections discuss the Domain and application engineering processes.

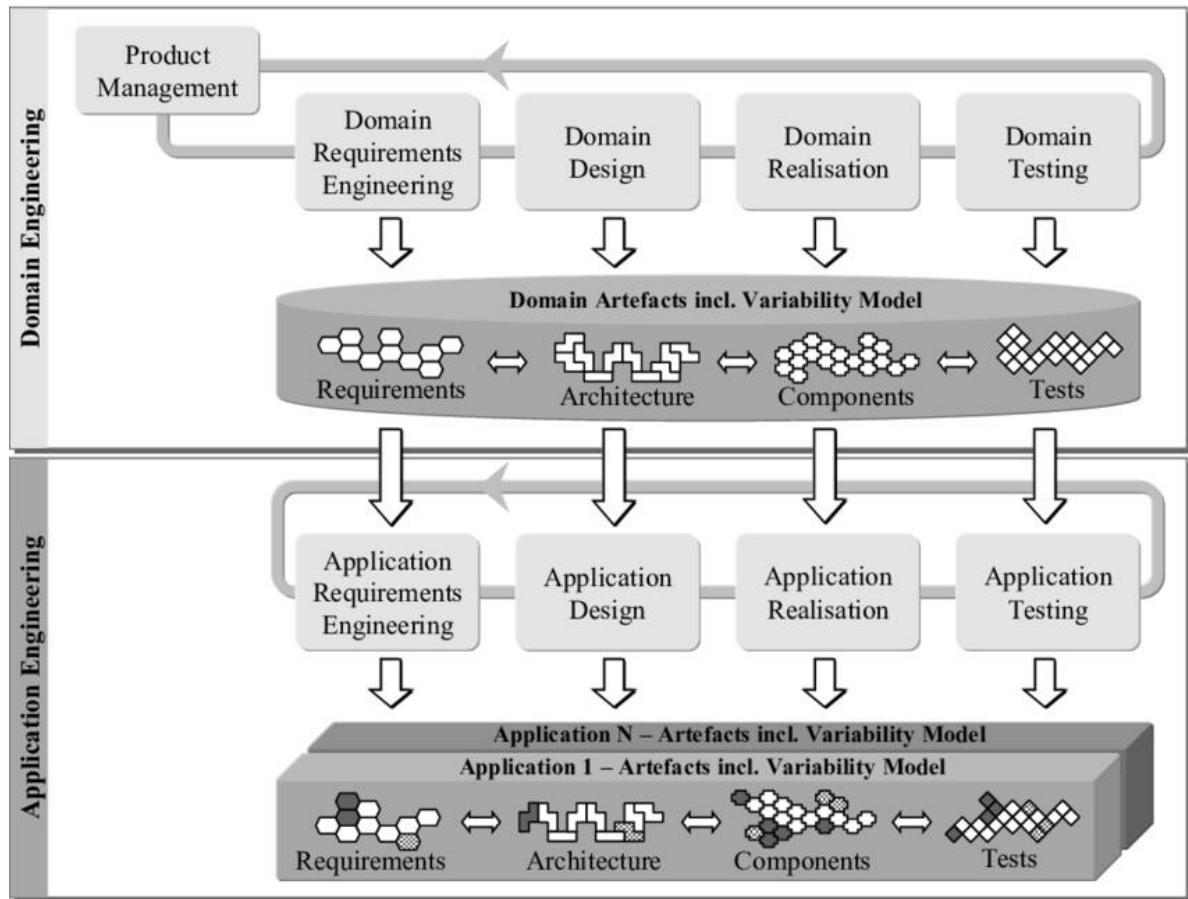


Figure 2. The software product line engineering framework (Pohl et al. 2005, p. 22)

### 3.4.2 Domain engineering

The goal of domain engineering is to develop reusable core assets for product derivation. To achieve this goal, domain engineering must identify and realize commonality and variability of the product line. (Pohl et al. 2005, p. 20)

Domain engineering encompasses five sub processes, which are discussed subsequently.

#### Product Management

Based on the business goals of a company, the product management sub process defines the scope of the product line a.k.a. the product portfolio. The product management process provides both a roadmap and a list of existing core assets, that can be systematically reused for product derivation. (Pohl et al. 2005, p. 25)

#### Domain Requirements Engineering

This process analysis requirements for of the entire product portfolio. The aim is to identify common and varying requirements between the products. The output of domain requirement engineering includes requirements and variability models for the entire product line. Application specific requirements engineering is still required. (Pohl et al. 2005, pp. 24–25)

In SPLE, variability can be defined as the ability of a core asset or any of its properties to change on purpose. A variation point is a representation of a varying core asset or its varying property. To identify variation points (Pohl et al. 2005, p. 60) propose posing the following questions, What does vary? Why does it vary? How does it vary? Applied to core assets the questions are:

- What core assets do vary or what properties of core asset vary?

Leads to precise variation point identification.

- Why does the core asset or its property vary?

This could for example be due to variation in other core assets or varying stakeholder requirements.

- How does the core asset or its property vary?

Posing this question leads to the identification of the different states, shapes or conditions a varying core asset can assume. In SPLE, these different assumable states are termed variants.

In SPLE, the domain engineering process designs and implements variability within core assets while the application engineering process exploits variability for derivation of concrete products.

SPLE identifies 4 types of variability namely, variability in time, variability in space, internal and external variability.

**Variability in time** is when different version of a core asset or any of its properties are valid at different times. Variability in time is used to accommodate core asset evolution with minimal effort and impact for the product line e.g. developers of an operating system define a variation point called authentication mechanism with a single variant password authentication. The authentication mechanism variation point later facilitates the introduction of a new variants e.g. fingerprint authentication.

**Variability in space** is the availability of a core asset in different version at the same time e.g. core asset developers simultaneously support password and fingerprint authentication as operating system authentication mechanisms.

**Internal variability** is variability that is not visible to customers e.g. developers of operating system used in health care implement a variation point ‘authentication mechanism’ with two variants, password- and fingerprint-authentication. If fingerprint authentication is selected, at

least one encryption standard must be chosen from 256-bit Advance Encryption Standard and 2048-bit RSA (Rivest-Shamir-Adleman).

**External variability** is that variability that is visible to the customer e.g. password- or fingerprint-authentication to an operating system.

Variability documentations ideally provide information about what the varying core asset is, why it exists, for whom it was intended and how it varies (the supported variants). Documenting variability is beneficial in that it improves communication, between developers as well as with customers. The documentation also improves decision making as it includes “rationales for including a particular variant”. (Pohl et al. 2005, p. 75)

In this thesis, variability is documented using both the orthogonal variability model (OVM), as proposed by (Pohl et al. 2005), and UML (Unified modeling language) (see Chapter 6, Figure 12). Figure 3 depicts the graphical notation for representing variability using the OVM. Figure 4 shows the authentication mechanism variation point using the OVM and UML.

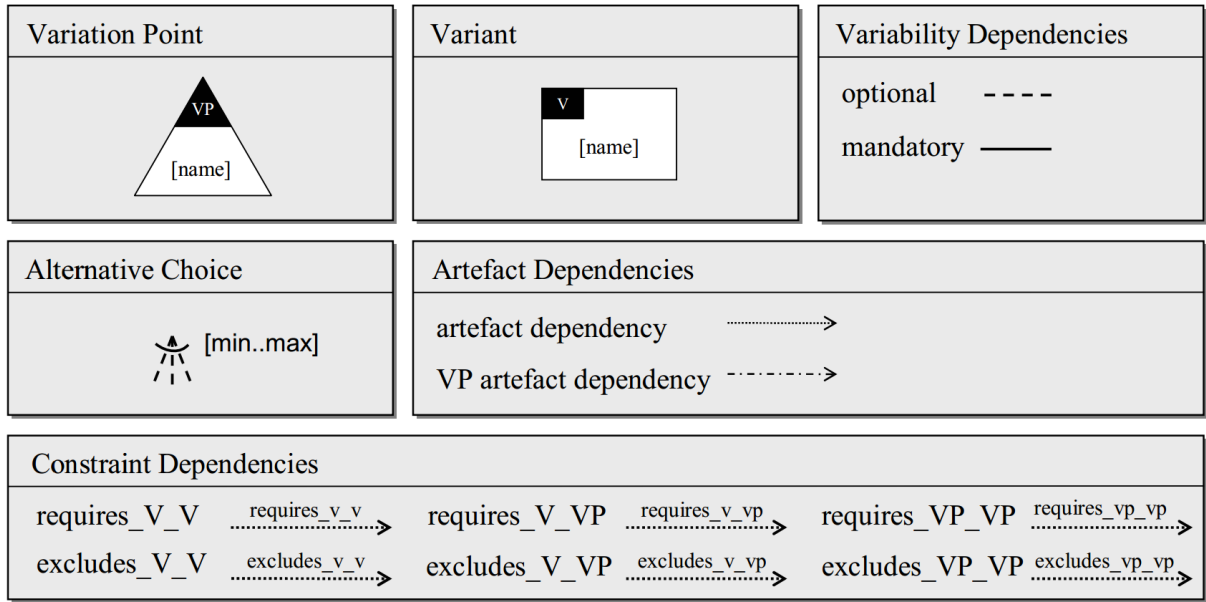


Figure 3 Graphical notation for variability models (Pohl et al. 2005, p. 85)



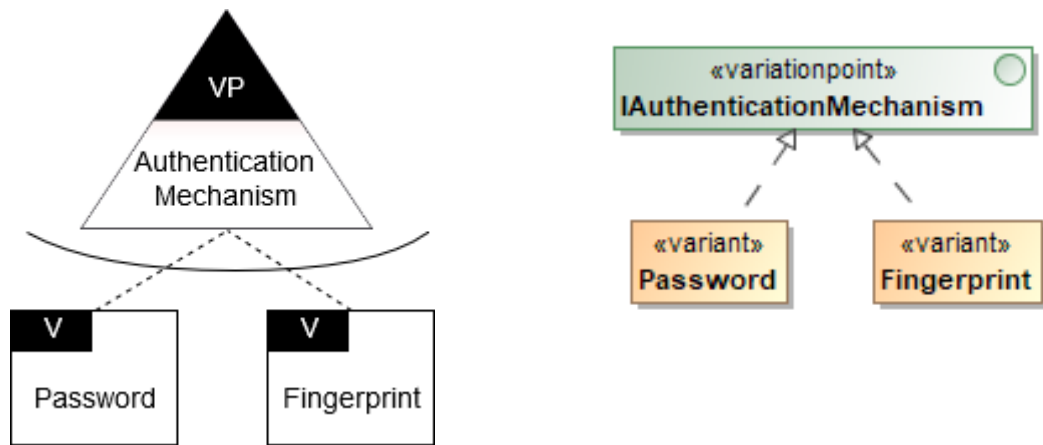


Figure 4 Authentication mechanism variation point using OVM vs UML

In the case of the UML representation, an interface is chosen as implementation for a variation point. The variants are classes that realize this interface. There are several techniques for implementing variability (Gacek and Anastasopoulos 2001).

### Domain Design

The goal of the domain design process is to define the product line reference architecture, based on the common and varying requirements from the domain requirements engineering process. (Pohl et al. 2005, p. 26)

### Domain Realization

The domain realization process implements core assets and configuration mechanisms that accommodate variability of the product line. Domain realization therefore provides the common platform used for product derivation and instantiation.

### Domain Testing

The domain testing process evaluates the reference architecture and developed core assets. Several strategies can be adapted for domain testing. In this thesis, we use the Sample Application Strategy (SAS) (Pohl et al. 2005, p. 284). The main advantage of SAS, is that it provides “an early validation of the software product line” (Pohl et al. 2005, p. 284). SAS involves the development of a sample application to validate core assets and supported variability (Pohl et al. 2005, p. 275).

Other domain testing strategies include, the Brute Force Strategy, Pure Application Strategy, Commonality and Reuse Strategy. For further information see (Pohl et al. 2005, pp. 271–280)

### **3.4.3 Application engineering**

Application engineering exploits commonality and variability of the product line for the derivation of specific products. The aim of application engineering is to maximize the reuse of developed core assets. Inputs to the application engineering process include but are not limited to the product lines scope, the product line reference architecture, the variability model and the production plan. (Clements and Northrop 2009, p. 37)

The sub processes of the application engineering process include Application Requirements Engineering, Application Design, Application Realization and Application Testing. (Pohl et al. 2005, p. 30)

This thesis focuses on the domain engineering process and therefore does not provide further detail on the application engineering process.

### 3.5 Cloud computing and Multitenancy

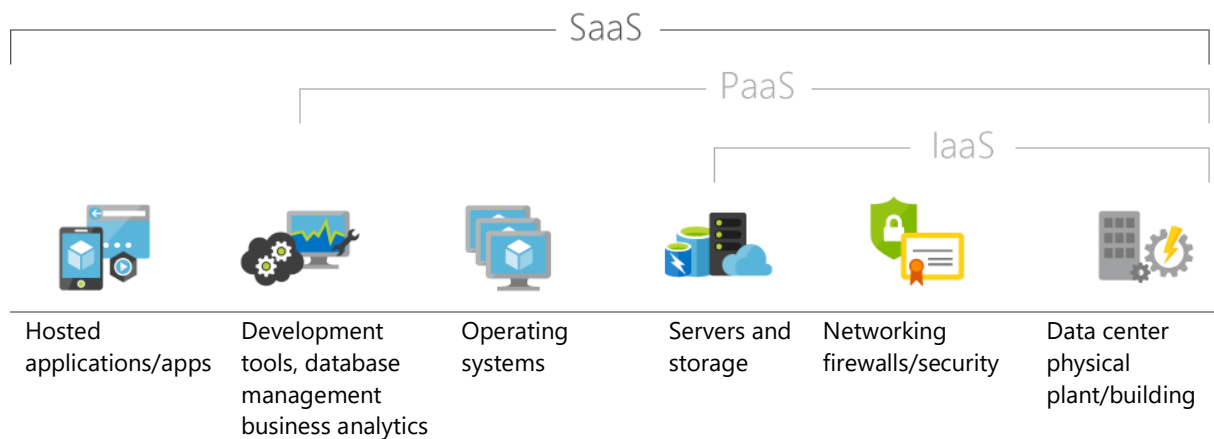


Figure 5. What is SaaS (Adapted from (What is SaaS? Software as a Service | Microsoft Azure))

The provision of computing services over the internet is known as cloud computing. Cloud providers are companies that deliver computing services e.g. Microsoft, Amazon etc. Figure 5, depicts the layers of cloud computing namely:

- **Infrastructure as a Service (IaaS):** This layer involves the use of a datacenter for the provision of computing via servers (aka. Virtualization), storage and networking services.
- **Platform as a Service (PaaS):** This layer involves the delivery of maintenance services for databases, server software and diverse development tools.
- **Software as a Service (SaaS):** This layer involves the provision of application hosting and maintenance (upgrades, security patching) services.

Multitenancy is an architectural approach for providing SaaS. In multitenant applications, multiple tenants share the same physical instance of an application. Physical instance refers to shared physical resources like Virtual Machines (VMs) (i.e. CPU, networking) and databases. A tenant refers to a group of users with common, access and privileges to a software instance. Due to reuse of physical resources, multitenant architectures enable reduced costs and are as a result beneficial for service providers. However, multitenancy requires complex application logic to isolate tenants. (Implementing a multi-tenant offering in Azure using CSP – Microsoft US Partner Community blog; What is SaaS? Software as a Service | Microsoft Azure; Cloud-10 Multi Tenancy and Physical Security - OWASP 2018; Multi-tenant SaaS patterns - Azure SQL Database 2018; Microsoft 2012; Mike Wasson 2018)

Different approaches (also referred to as tenancy models) exist for storing tenant data. In the single-tenancy model, data is stored in a separate database for each tenant (see Figure 6). In the

multi-tenancy model, data from all tenants is stored in the same database (see Figure 7). Hybrid tenancy models also exist that combine both afore mentioned models.

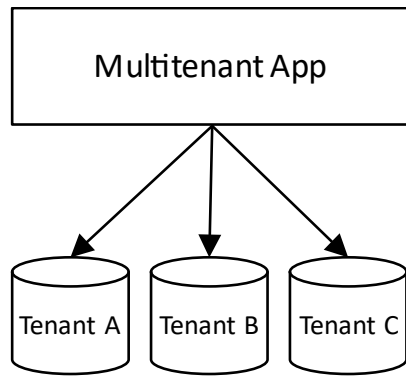


Figure 6. Multitenant App with database per tenant

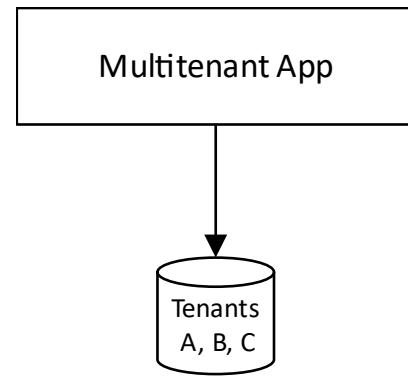


Figure 7. Multitenant App with multi-tenant database

To assure that tenants are properly isolated, the database in the multitenancy model must be extended with a column or more for tenant identification. Later versions of the Microsoft SQL database support tenant isolation in multitenancy models via the row-level security mechanism (see (Row-Level Security 2018)). The EHR with integrated CDS-System applications, developed in this work, are offered to customers as SaaS and are hosted either on Microsoft Azure or on Microsoft Internet Information Services (IIS) servers. This thesis adopts the multi-tenancy model and ensures tenant data isolation by leveraging the row-level security mechanism of the Microsoft SQL database. Because of adopting a dynamic SPL approach, tenants share a single domain instance. Via tenant configurations, reusable software components are used to specify the tenant context, thus providing a tenant specific view on the instance. Available software components can be used in more than one tenant contexts.

## 4 CDS Information

This section aims to address questions related to the information needs of physicians at the point of care, namely:

- What are the information needs of physicians at the point of care?
- What is the information seeking behavior of physicians?
- What web-based medical information retrieval resources are commonly used amongst physicians?

### 4.1 Physicians' information needs

Many studies have been conducted to identify the information needs of physicians at the point of care (Clarke et al. 2013). Table 2 shows the result of a literature review, conducted to identify the needs of physicians and nurses at the point of care.

Study	Diagnosis	Drug	Treatment	Prognosis	Epidemiology	Etiology
Allen et al., 2003	x	x				
Alper et al., 2005	x		x	x	x	x
Bergus and Emerson, 2005	x		x	x		
Cao et al., 2010	x	x	x	x	x	x
Chan and Stieda, 2011		x	x			
Chen et al., 2006		x				
Cheng, 2004	x	x	x	x		
Cimino, 2006		x				
Cimino, 2007	x	x				
Cogdill et al., 2000	x	x			x	x
Cucina et al., 2001	x	x		x		
Cullen, 2002	x	x				
Del Fiol and Haug, 2008		x				
Ebell and White, 2003	x	x	x		x	
Fozi et al., 2000	x	x	x	x		
Gonzalez-Gonzalez et al., 2007	x		x		x	
Gorman et al., 2004			x			
Green et al., 2000	x		x	x		x
Lauscher et al., 2008					x	
Magrabi et al., 2005	x	x	x			x
Maviglia et al., 2006		x				
Schilling et al., 2005	x		x	x	x	x
Schwartz et al., 2003	x		x	x	x	
Swinglehurst et al., 2001	x		x	x		x
<b>Total (24)</b>	<b>17</b>	<b>15</b>	<b>14</b>	<b>10</b>	<b>8</b>	<b>7</b>

Table 2. List of information needs of physicians and nurses commonly mentioned in the review of literature (Clarke et al. 2013)

(Clarke et al. 2013) assert that physicians' information needs at the point of care are associated with diagnosing health conditions, finding best treatment methods (therapies) and drug prescriptions. Physicians' needs for prognosis, epidemiology and etiology information serve the purpose of confirming diagnosis and therapies (Clarke et al. 2013).

This thesis proposes CDS-Services, to meet the information needs of physicians at the point of care, namely:

<b>CDS-Service</b>	<b>Targeted information need</b>
Drug interaction service	Drug
Drug adverse event service	Drug
Clinical trial service	Drug, treatment, prognosis, epidemiology, Etiology
Evidence-based medical guideline service	Treatment
Literature service	Drug, treatment, prognosis, epidemiology, etiology

Table 3 CDS-System information services and addressed information needs

As can be seen in Table 2, none of the proposed CDS-Services assist physicians in diagnosing patients' issues. The services are rather for post diagnosis scenarios.

## 4.2 Information retrieval resources for CDS content

As shown in Table 4, considerable effort has been performed, to identify the information seeking behavior of clinicians. According to the findings in (Clarke et al. 2013), the most consulted resource is the internet. Regarding these findings, the question to pose is, what web-based medical information retrieval resources are commonly used amongst physicians?

In (Maggio et al. 2014), PubMed and UpToDate are identified as belonging to widely used Web-based information retrieval resources. The work in (Johannes Idelhauser 2016), identified further web-based information retrieval resources shown in Appendix A,B,C and D.

Study	Internet	Textbooks	Journals	Colleague	Drug Compendium	Professional websites	Medical Libraries
Adams et al., 2007	x	x		x			
Alghanim, 2011	x	x	x	x			
Alper et al., 2005	x	x	x			x	
Andrews et al., 2005	x	x	x		x	x	
Bennett et al., 2005	x		x	x		x	
Bertulis and Cheeseborough, 2008	x	x	x				
Cimino et al., 2003	x	x	x		x		
Cullen et al., 2011	x		x				x
Cullen, 2002	x	x		x			x
Ebell and White, 2003	x	x	x	x	x		
Edson et al., 2010	x	x	x	x			
Fozi et al., 2000	x	x	x				x
Gonzalez-Gonzalez et al., 2007	x	x	x	x	x		
Gorman et al., 2004	x	x	x	x	x	x	
Green et al., 2000		x	x	x			
McKibbin and Fridsma, 2006	x						
Norbert and Lwoga, 2012		x	x	x		x	x
Perley et al., 2007				x			
Randell et al., 2009	x	x					
Schilling et al., 2005	x	x	x	x	x	x	
Schwartz et al., 2003	x						
Turner et al., 2008		x		x			
<b>Total (22)</b>	<b>18</b>	<b>17</b>	<b>15</b>	<b>13</b>	<b>6</b>	<b>6</b>	<b>4</b>

Table 4. List of information sources utilized by physician and nurses (Clarke et al. 2013)

In this thesis, information retrieval resources are selected based on the following criteria:

- availability of an API
- freely accessible without registration and

- volume of information

Table 5 shows the list of selected information retrieval resources used in this thesis.

<b>CDS-Service</b>	<b>Information retrieval resource</b>	<b>Updates</b>
Drug interaction service	RxNav Drug Interaction API (Sources: DrugBank and ONCHigh)	Monthly
Drug adverse event service	OpenFDA Drug adverse event API	Quarterly
Clinical trial service	ClinicalTrials.gov	Daily
Evidence-based medical guideline service	National Comprehensive Cancer Network (NCCN) Guidelines	Updated at NCCN's discretion
Literature service	PubMed	Daily

Table 5 CDS-Services, selected information retrieval resources



# 5 Architectures for Clinical Decision Support Systems

This chapter aims to address the following questions:

- What architectural approaches exist for implementing CDS-Systems?
- Which architectural approaches support modularity, flexibility and reusability of CDS-Systems?

To answer these questions, a literature review was performed to identify relevant literature. Found publications include (Kawamoto et al. 2010; Loya et al. 2014; Wright and Sittig 2008; Greenes 2014a; Kawamoto et al. 2014; Rodriguez-Loya and Kawamoto 2016; Greenes 2014b).

## 5.1 Evolution of CDS architectures

Many approaches have been adopted for building CDS-Systems. All identified studies agree on the fact that, there has been an evolution from tightly coupled monolithic CDS-System architectures to loosely coupled service-oriented architectures.

Beginning from the year 2005, (Wright and Sittig 2008) identify four evolutionary approaches for CDS-Systems. Figure 8 shows the different approaches and systems developed based on these approaches. Advantages and disadvantages for each approach are discussed subsequently.

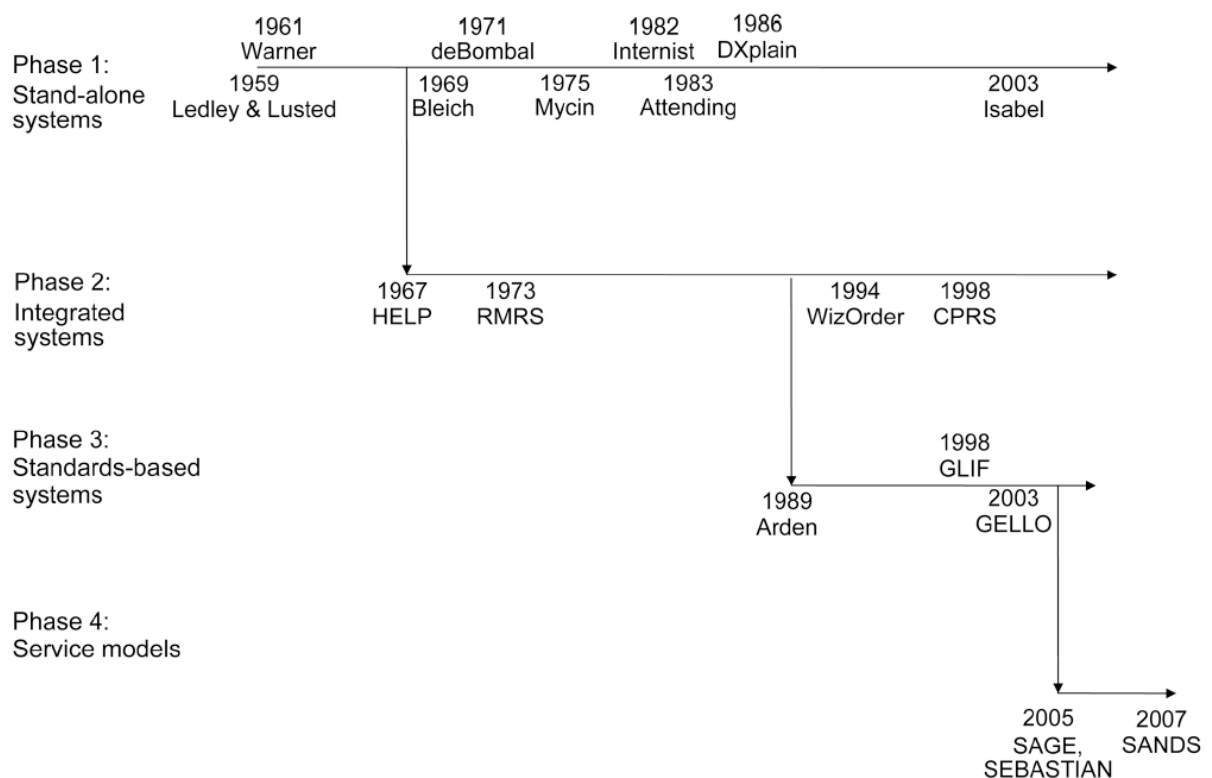


Figure 8 A schematic drawing of the four-phase model for clinical decision support (Wright and Sittig 2008)

### 5.1.1 Stand-alone CDS-Systems

Stand-alone CDS-Systems function independently of the EHR.

Advantages	Disadvantages
The effort required for developing stand-alone systems is manageable, given the existence of domain knowledge and software engineering skills.	These systems require double entry of patient data, first into the EHR, then into the CDS-System. This is both error prone and time consuming.
This approach does not require the specification of standards.	Stand-alone systems cannot provide clinical decision proactively.
Stand-alone systems are easy to deploy and share.	

Table 6 Advantages and disadvantages of Stand-alone CDS-Systems (Wright and Sittig 2008)

### 5.1.2 Integrated CDS-Systems

These systems integrate CDS into EHRs, however, via tightly coupled implementations.

Advantages	Disadvantages
Integrated CDS-System do not require data entry and therefore solve the problems faced with stand-alone systems.	Sharing CDS content across organizations requires that the involved parties use the same EHRs.
Integrated systems enable proactive decision support.	Integrated systems encourage mixing EHR code with CDS code. In cases where code for both systems is mixed, maintaining and updating existing code requires much effort.
	Marketing integrated systems is difficult due to clinicians' reluctance to replace existing systems. Medical institutes will rather tend to extend existing systems to provide CDS. A common approach is to use info buttons (see (Del Fiol et al. 2014)).

Table 7 Advantages and disadvantages of tightly coupled integrated CDS-Systems (Wright and Sittig 2008)

### 5.1.3 Standards-based CDS-Systems

This approach leverages standard specifications for encoding, representing, storing and sharing CDS content.

Advantages	Disadvantages
Enable the sharing of CDS content across organizations.	The existence of manifold standards impedes CDS content sharing, since this requires the implementation of mapping mechanisms.
Enforce separation of EHR application code from CDS-System code. This enhances code maintainability.	Concepts without the scope of a given standard are not supported. Standards therefore limit users to standard scoped concepts, e.g. the Arden Syntax only supports CDS in the form of event-driven based alerts and reminders.

Table 8 Advantages and disadvantages of standards-based CDS-Systems (Wright and Sittig 2008)

#### 5.1.4 Service oriented CDS-Systems

These CDS-Systems adopt loosely coupled software design principles, such as SOA. SOA organizes complex system functionalities into smaller, reusable modules called services, that are accessible via standardized Application Programming Interfaces (APIs) (Advancing Open Standards for the Information Society (OASIS). 2006), (Kawamoto et al. 2014). Benefits of SOA include flexibility, modularity, reusability (also of legacy applications), maintainability, and interoperability (SOA Features and Benefits; Rodriguez-Loya and Kawamoto 2016). SOA also promises reduction in implementation time and costs, then services are developed mainly by integration of well-defined services (Welch et al. 2014b). Using SOA, a CDS-System can be implemented by composing various CDS-Services that provide desired information and functionality. Furthermore, SOA-based CDS-System approaches enable distributed (across different vendors) development of CDS functionalities, enforce separation of concerns and accommodate centralized CDS knowledge management and sharing (Kawamoto et al. 2010).

The SOA-based approach best suits the requirements for rapid implementation, flexibility, modularity, maintainability, evolvability and portability of the CDS-Services, and is therefore chosen as architectural approach for this thesis. Studies, also reveal a wide adoption of the SOA-based approach for CDS (Loya et al. 2014).

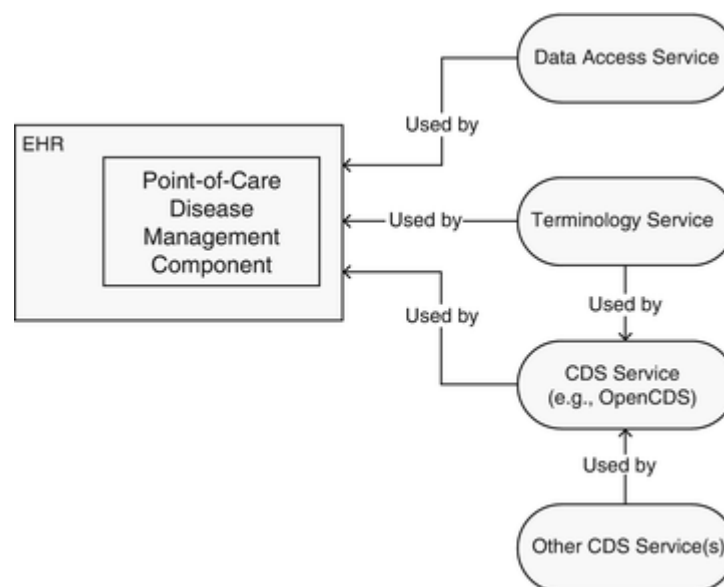


Figure 9 Sample CDS system architecture (Adapted from (Rodriguez-Loya and Kawamoto 2016))

Figure 9 depicts a sample of a SOA-based CDS-System architecture. This shows how a complex EHR system can be realized using various services, namely, a Data Access service for retrieving patient specific data, a Terminology Service and a CDS-Service. The CDS-Service

in this sample system runs as a web service and is accessible via an API. The CDS web service might also use other web-based CDS-Services.

## 5.2 CDS-System design components

As aforementioned, the goal of this thesis is to build CDS-Services that enable easy integration of CDS into EHR applications. This section discusses the architectural components of EHR applications with integrated CDS.

Generally, the architecture of EHR applications with integrated CDS is constituted of 5 components as shown in Figure 10, namely:

1. The Execution engine: Processes input e.g. patient data (in form of prescribed medications, diagnosed diseases and patient history), to produce recommendations.
  2. The Knowledge base: Holds information used by the execution engine for reasoning. This component is optional as CDS-System may be implemented such that they use external knowledge bases.
  3. Information model: Holds data format specifications for expected input.
  4. Result specification: Returns computed results to the invoking application.
  5. Application environment: Invokes the CDS-System and manages user interaction with the CDS-System. For EHRs integrating a CDS-System it is common that the application environment retrieves and forwards patient data when invoking the CDS-System.
- (Greenes 2014b; Kawamoto et al. 2014)

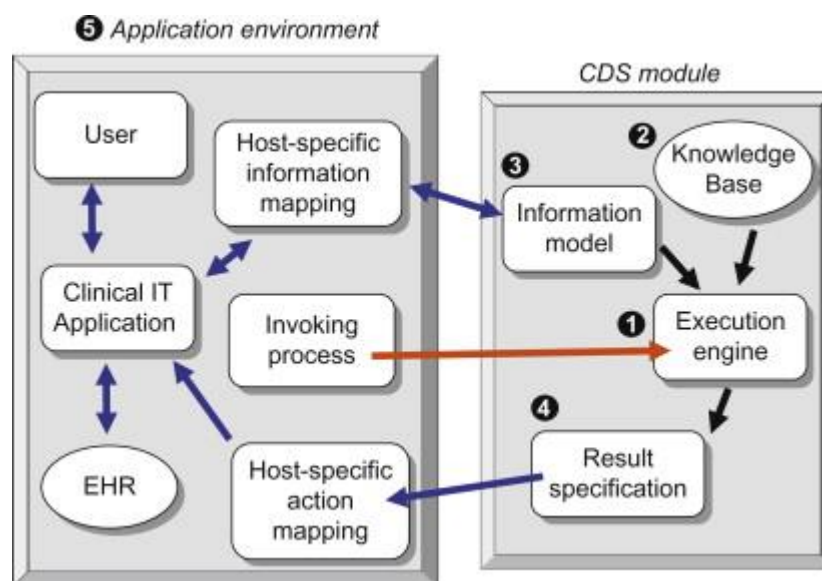


Figure 10 A conceptual model of CDS design components and their interactions with the host application environment. (Greenes 2014b)

As shown in Figure 10, the process of retrieving CDS information, is initiated by an invoking process within the application environment. In most cases, the invoking process passes patient-specific data to the Execution engine of the CDS module (CDS-System). The Execution engine understands the input, thanks to the data format specifications in the Information model component. Using knowledge from the Knowledge Base component, the execution engine computes the received input and generates an output, that is forwarded to the Invoking application process via the Result specification component. (Greenes 2014b)

Developing an integrated CDS-System can thus be achieved by systematic composition of the above-mentioned components.

In summary, flexibility, modularity and portability of CDS-Systems can be achieved by:

- Cleanly separating the CDS-System from the invoking EHR application and communicating with it via APIs. (Greenes 2014b)
- Separating the code of the execution engine from the code of the EHR application and from other parts of the CDS-System as well. This enables separate development, maintaining and enhancing of the execution engine. Secondly, “this provides flexibility and portability, in that, the execution engine can be recoded and reimplemented in different platforms independent of other CDS parts, and can even be embodied in external services [...]” (Greenes 2014b)
- Using a standardized information model e.g. the HL7 RIM. This enables the CDS-Service to be used in a variety of settings, “[...] e.g. interactively with a user as well as in background mode, retrieving data from the EHR, and in more than one platform and system environment.” (Greenes 2014b)
- Separating the result specification component from the rest of the CDS-System. This provides for flexibility in the mode of interaction with the CDS-System, therefore facilitating portability of the CDS-System. The result of the CDS-System could for example “be provided in real-time in interactive applications, in the background in alert/reminder usages or in batch mode in the production of reports or summaries.” (Greenes 2014b)

## 6 SOA-based CDS-Services

The aim of this thesis is to integrate CDS into EHR applications of a product line via CDS-Services. The CDS-Services must therefore be reusable, a quality that promotes them to core assets of the product line. The terms CDS-Services and CDS core assets are therefore used synonymously (see Figure 12). To achieve reusability, the CDS-Services must be portable and independent of the EHR application into which they are integrated.

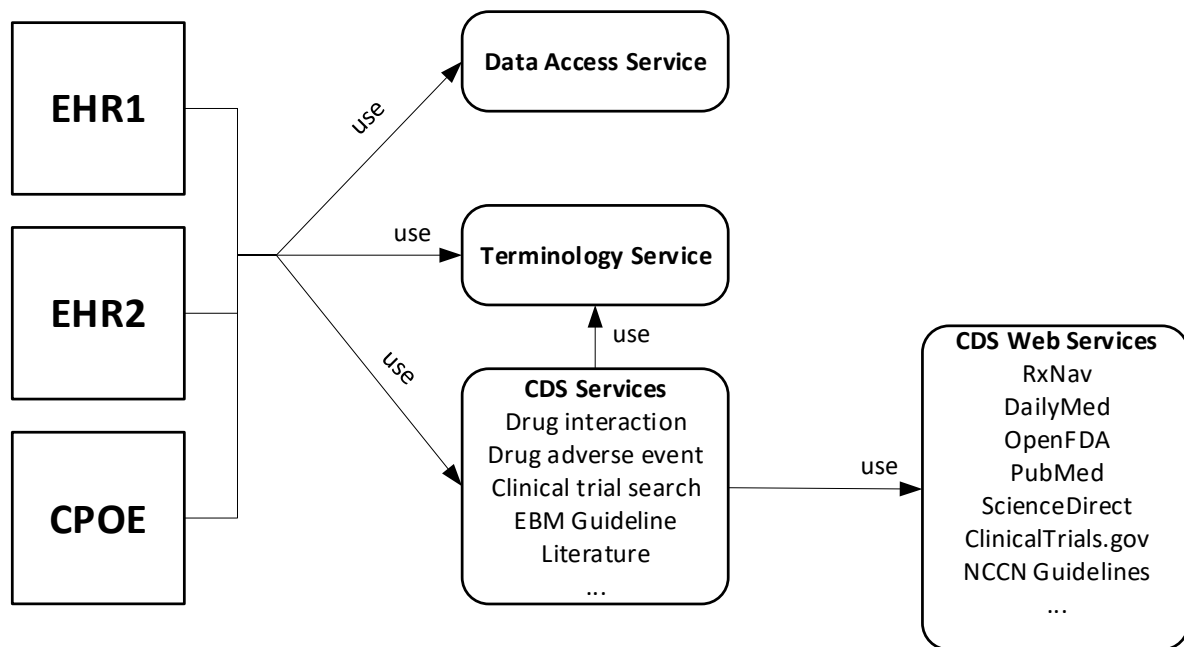


Figure 11 CDS-Services in a SOA-based Health information systems environment

Figure 11 depicts the proposed concept. The concept adopts a SOA based approach to provide CDS Services to several EHR applications (EHR1 and EHR2) but also to other health information systems e.g. a CPOE. The functionalities of the CDS services are exposed via APIs. The same is true for the functionalities of the Data Access and Terminology Services, that provide access to patient data and medical vocabulary respectively. The CDS services may either provide locally implemented CDS functionalities or access external medical information retrieval resources (CDS Web Services) via their APIs.

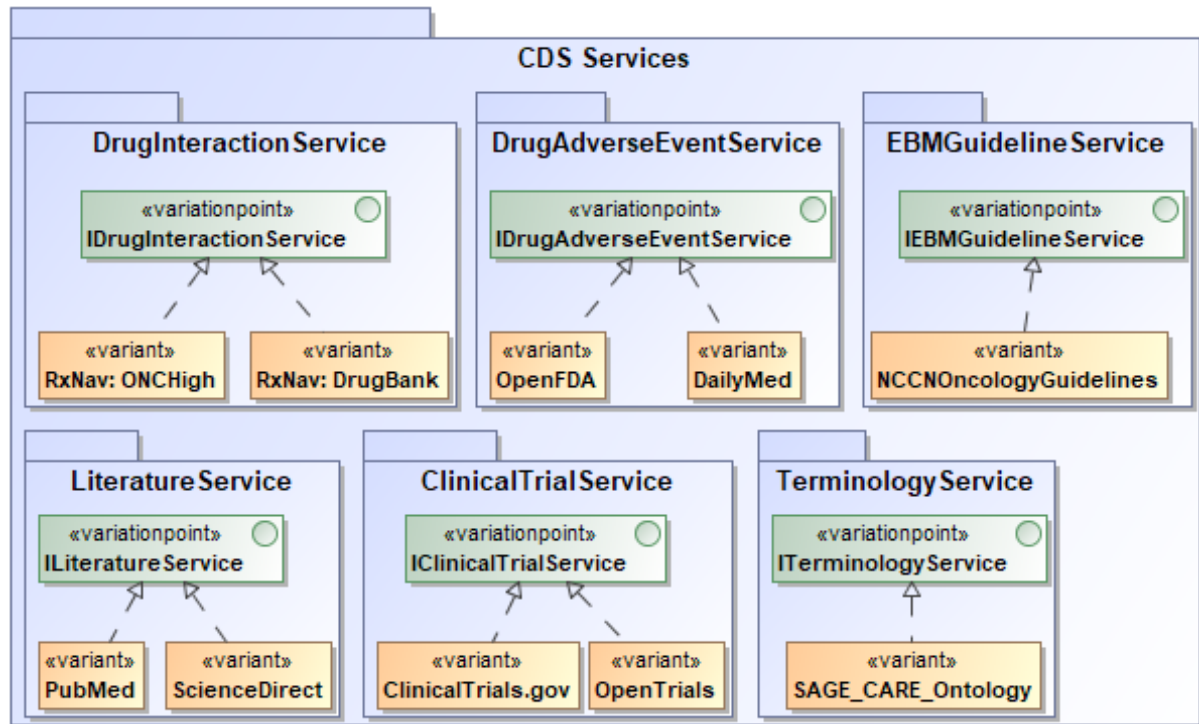


Figure 12 CDS-Services architecture

Figure 12 shows the architecture for the CDS-Services. Each CDS-Service contains an interface that serves as a variation point to facilitate the introduction of new CDS Web Services. Variants are classes that realize the interface and implement RESTful client logic for retrieving CDS information from selected CDS Web Services.

To introduce a new information retrieval resource (variant) for any CDS-Service, the CDS-Services developer:

- creates a class that implements the selected service interface
- programs RESTful client logic to query and retrieve CDS information from a selected CDS web service

To introduce a new CDS-Service, the CDS-Service developer:

- creates an interface and defines the methods that must be implemented by variants
- follows the afore mentioned steps for introducing a new information retrieval resource

An important issue to be addressed when using interfaces, is how to eliminate code duplication since each variant must implement all methods defined in the interface. To address this issue, the CDS-Service developer shall implement utility classes with static methods.

The CDS-Services are therefore loosely coupled to the EHR application into which they are integrated, resulting in flexibility and portability. The CDS-Services have different patient-data

demands for retrieving CDS recommendations. The interfaces therefore vary accordingly. Table 9 shows the interfaces and their specifications<sup>2</sup>.

CDS-Service interface	CDS-Service Methods
IClinicalTrialService	Uri GetClinicalTrialQueryLink (ClinicalTrialQuery query);
IDrugAdverseEventsService	ICollection<DrugAdverseEvents> FindDrugAdverseEvents (IList<string>patientMedications, int limit)
IDrugInteractionService	IList<DrugInteraction> FindDrugInteractions (IList<string> medications);
IEBMGuidelineService	EBMGuideline SearchGuidelines (string issueName, ClinicalStage? clinicalStage, string medicalTerms);
ILiteratureService	IList<string> SearchForLiteratureIds (IList<SearchField> ehrFields, int maxResultSize);
	IList<Literature> GetLiteratureByIds (IEnumerable<string> publicationIds);
ITerminologyService	IList<Term> Autosuggest (TerminologyAutosuggestData autosuggestData);

Table 9 CDS-Services interface specifications

To use these services, the application developer shall first retrieve data from the EHR and call the API methods passing patient-specific data in the awaited formats (see Listing 1).

```
private static IDrugInteractionService rxnormDIService = new
RxNormDrugInteractionService();

//Retrieve patient medications from EHR store

return rxnormDIService.FindDrugInteractions(patientMedications);
```

Listing 1 Using the drug interaction service

The following chapters discuss the CDS-Services in detail.

## 6.1 Drug interaction service

The aim of this service is to provide drug-drug interaction information to physicians at the point of care. For this purpose, the drug interaction service requires a list of the patient's medications, prescribed as well as not yet prescribed. The not yet prescribed drugs are important for checking drug interactions between already prescribed and not yet prescribed drugs. The indication of such interactions calls for action from the physician. Listing 2 shows the IDrugInteraction interface, while Listing 3 depicts the response data format, namely, DrugInteraction. The flow of logic for the drug interaction service is depicted using a sequence diagram in Figure 40 of Appendix E.

<sup>2</sup> The programming language used for the Listings is C#.



```
public interface IDrugInteractionService
{
    IList<DrugInteraction> FindDrugInteractions(IList<string> medications);
}
```

Listing 2 Drug interaction service interface - IDrugInteractionService

```
public class DrugInteraction
{
    public string Description { get; set; }

    public string Severity { get; set; }

    public List<string> InteractingMedications { get; set; }

    public string Comment { get; set; }
}
```

Listing 3 Drug interaction service interface response data format – DrugInteraction

## 6.2 Drug adverse events service

The aim of this service is to retrieve information of drug related adverse events for the patient's prescribed and not yet prescribed drugs. The service interface specification is shown in Listing 4. Listing 5 and Listing 6, show the specification of the service's response data format. Figure 41 of Appendix F depicts the flow of logic for this service.

```
public interface IDrugAdverseEventsService
{
    ICollection<DrugAdverseEvents> FindDrugAdverseEvents(IList<string>
        patientMedications, int limit = 1);

    DrugAdverseEvents FindDrugAdverseEvents(string medication, int limit = 1);
}
```

Listing 4 Drug adverse events service interface – IDrugAdverseEventsService

```
public class DrugAdverseEvents
{
    public string Medication { get; set; }
    public ICollection<AdverseEvent> AdverseEvents { get; set; }
}
```

Listing 5 Drug adverse events service response data format - DrugAdverseEvents

```

public class AdverseEvent
{
    /// <summary>
    /// Name of the reported adverse event
    /// </summary>
    public string Term { get; set; }
    /// <summary>
    /// Number of reports
    /// </summary>
    public int Count { get; set; }
}

```

Listing 6 Drug adverse events service response data format – AdverseEvent

## 6.3 Clinical trial service

The aim of the clinical trial service is to provide information on clinical trials that may be worth considering for the patient. Listing 7 shows the interface specification of the clinical trial service.

```

public interface IClinicalTrialService
{
    Uri GetClinicalTrialQueryLink(ClinicalTrialQuery query);
}

```

Listing 7 Clinical trial service interface - IClinicalTrialService

```

public class ClinicalTrialQuery
{
    public string Condition { get; set; }
    public string Term { get; set; }
    public string Country { get; set; }
    public string State { get; set; }
    public string City { get; set; }
    public int Distance { get; set; }
    public int? Age { get; set; }
    public Gender? Gender { get; set; }
    public DateTime StudyStartFrom { get; set; }
    public DateTime StudyStartTo { get; set; }
}

```

Listing 8 Clinical trial service parameter data format – ClinicalTrialQuery

Listing 8 shows the specification of the ClinicalTrialQuery parameter, that holds patient-specific data from the EHR and Listing 9 shows the specification of the Gender data type.

```

public enum Gender
{
    [Description("Female")]
    FEMALE,

    [Description("Male")]
    MALE,

    [Description("Other")]
    OTHER,

    [Description("All")]
    ALL,
}

```

Listing 9 Specification of the gender data type

## 6.4 Literature service

The aim of this service is to retrieve relevant (patient-specific) medical publications. Listing 10 shows the Literature service interface. The query parameter is a list of SearchField objects. As shown in Listing 11 and Listing 12, this object provides an EHR field mapping (in the format, field name : field value). The return data type is shown in Listing 13.

```

public interface ILiteratureService
{
    IList<string> SearchForLiteratureIds(IList<SearchField> ehrFields,
        int limit);
    IList<Literature> GetLiteratureByIds(IEnumerable<string> publicationIds);
}

```

Listing 10 Literature service interface – ILiteratureService

```

[Serializable]
public class SearchField
{
    private string _value;

    public SearchField(EhrField fieldName, string value)
    {
        FieldName = fieldName;
        Value = value;
    }

    public EhrField FieldName { get; set; }

    public string Value
    {
        get { return _value; }
        set { _value = value.ToLower(); }
    }
}

```

Listing 11 EHR mapping for literature service – SearchField

```

public enum EhrField
{
    Age,
    BiopsyType,
    ClinicalStage,
    Comorbidity,
    Gender,
    PositiveGene,
    NegativeGene,
    IssueType,
    LesionSite,
    LesionType,
    Medication,
    Other,
    Procedures,
    Tstage,
    Ulceration
}

```

Listing 12 EhrField data type for EHR mapping in literature service

```

public class Literature : IEquatable<Literature>
{
    [JsonProperty("id")]
    public string DocumentId { get; set; }

    [JsonProperty("title")]
    public string ArticleTitle { get; set; }

    [JsonIgnore]
    public string AbstractText { get; set; }

    [JsonProperty("conclusion")]
    public string AbstractConclusion { get; set; }

    public IList<string> PublicationType { get; set; } = new List<string>();

    [JsonIgnore]
    public string PublicationYear { get; set; }

    [JsonIgnore]
    public string PublicationMonth { get; set; }

    [JsonIgnore]
    public string JournalTitleAbbreviation { get; set; }

    [JsonIgnore]
    public string JournalVolume { get; set; }

    [JsonIgnore]
    public string JournalIssue { get; set; }

    public string JournalString { get; set; }

    [JsonIgnore]
    public IList<string> MeshTerms { get; set; } = new List<string>();

    public bool IsReview { get; set; } = false;
    public bool IsClinicalTrial { get; set; } = false;
    public bool IsCaseReport { get; set; } = false;
}

```

Listing 13 Literature service response data format - Literature

The simplified flow of logic for this service is as follows: The EHR application developer, calls the `SearchForLiteratureIds` method passing patient-specific data to the service. The service queries a selected service e.g. PubMed and retrieves the identifiers (Ids) of fitting medical publications. The application developer then calls the `GetLiteratureByIds` method to retrieve medical publications by their Ids and returns these in the format of a Literature object to the EHR application developer (see Listing 13). For more information on the literature service see (Ulrich Beez 2015; Johannes Idelhauser 2016).

## 6.5 Terminology service

The terminology service is used for automatically suggesting a list of medical terms that match the user's input in selected (by EHR application developers) free text fields. Terms are retrieved from a term store implemented in the scope of the SAGE-CARE project using a Solr<sup>3</sup> Apache Server. Terms are classified into 6 semantic categories, namely, Activity, Anatomy, Disease, Gene, Medication and Symptom.

```
public interface ITerminologyService
{
    IList<Term> Autosuggest(TerminologyAutoSuggestData autosuggestData);
}
```

Listing 14 Terminology service interface - ITerminologyService

---

<sup>3</sup> Apache Solr is an open source enterprise search server based on the Apache Lucene Java search library, with XML/HTTP and JSON APIs, hit highlighting, faceted search, caching, replication, and a web administration interface. See <http://lucene.apache.org/solr> for more information.

<sup>3</sup> MS Services are synonymous to EHR Services.

```

public class Term : IEquatable<Term>
{
    public string SourceId { get; set; }
    public string TermId { get; set; }
    public string Label { get; set; }

    [JsonConverter(typeof(StringEnumConverter))]
    public TermCategory Category { get; set; }

    [JsonConverter(typeof(StringEnumConverter))]
    public TermSource Source { get; set; }

    public string Definition { get; set; }

    public IList<string> Broader { get; set; }

    public ISet<string> Synonyms { get; set; }

    [NotMapped]
    public double Score { get; set; }

    public double Rating(){...}
    ...
}

```

Listing 15 Terminology service response data format -Term

Listing 14 shows the `ITerminology` interface. The `Autosuggest` method requires a `TerminologyAutosuggestData` object that contains the term to be autocompleted and some other information shown and described in Listing 16. The Service returns a List of Term objects (see Listing 15). Listing 16 shows the specification of the `TerminologyAutosuggestData` class, used for querying the Terminology service.

```

public class TerminologyAutosuggestData
{
    /// <summary>
    /// Text input to be autocompleted
    /// </summary>
    public string Text { get; set; }

    /// <summary>
    /// Number of Terms to be returned
    /// </summary>
    public int Count { get; set; }

    /// <summary>
    /// String conforming to enum TerminologyService.OrderingType:
    /// - "ByScoreOnly": terms ordered according to term relevance only
    /// - "CategoryClusters": terms ordered according to cluster relevance only
    /// - "CategoryDiversity" (default): combines term and cluster relevance
    /// </summary>
    public string OrderingType { get; set; }

    /// <summary>
    /// There are 6 semantic categories of terms:
    /// symptom, disease, anatomy, gene, medication, and activity
    /// In an autosuggest request, retrieves terms of 1/more semantic categories.
    /// To exclude a semantic category from a query, set its weight to 0
    /// If a semantic category is to be considered its weight must be > 0, e.g. 1
    /// Priorities can be indicated using differing weights. E.g., for twice more
    /// medications than diseases, set MedicationWeight=2 and DiseaseWeight=1
    /// </summary>
    public float MedicationWeight { get; set; }
    public float ActivityWeight { get; set; }
    public float DiseaseWeight { get; set; }
    public float GeneWeight { get; set; }
    public float SymptomWeight { get; set; }
    public float AnatomyWeight { get; set; }
}

```

Listing 16 Terminology service parameter data format

(Beez et al. 2015; Ulrich Beez 2015) discuss the Literature service in more detail.

The following chapters discuss the prototyping of the proposed concept using the SAGE-CARE product line of EHR applications as use case.

## **7 The SAGE-CARE product line use case**

In the scope of previous work at the SAGE-CARE project, core assets have been developed that enable the derivation of EHR applications targeting different medical specialties (Patrick Spitzer 2016). This thesis aims to integrate CDS into EHR applications of the SAGE-CARE product line using the CDS-Services discussed in Chapter 6. The following section discuss the different domain engineering subprocesses for generating core assets of the product line. Core assets are those artefacts that are systematically reused to build concrete products, in this case EHR applications with integrated CDS-Services. The following chapters also discuss the commonalities and variabilities of the CDS-Services.

### **7.1 Product Management**

Based on the business goals of a company, product management defines the products within the scope of the product line (Pohl et al. 2005). The defined goal is to integrate CDS into EHR applications of the SAGE-CARE product line via CDS-Services. The output of the product management sub process is a product roadmap and a list of previous products that may offer reusable assets for the development of new products. For this thesis, the product list consists of the Simplicity-MDT EHR application from NSilico Lifescience Ltd. (NSilico), the Melanoma EHR application and other EHR core assets developed at the UASD.

### **7.2 Domain requirements engineering**

Software Product Line Engineering involves the building of software intensive systems from a common set of core assets. The core assets can be combined in different ways, so the derived products meet the needs of individual customers. This section aims at addressing how derived CDS-Systems shall support customization, to meet the needs of different physicians.

#### **7.2.1 Commonality analysis**

This section textually documents the requirements (features) that are common to all derived CDS-Systems.

##### **Intended user**

Dependent of the intended user's role (e.g. physician, nurse, pharmacist, radiologists, laboratory technicians, patients etc.), the approach for providing CDS and the type of information provided may vary. The CDS-Systems are intended for physicians.



## Purpose and key methodologies

All derived CDS-Systems shall pursue the purposes of answering physicians' questions, optimizing clinical process workflow and monitoring of actions at the point of care. Table 10, describes these purposes in detail.

Purpose	Description	Key methodologies
Answering questions	The CDS-System shall use "information brokers" to map all retrieved settings-, context- and patient-specific information from the user's query to formats understood by external search tools and knowledge bases. The response from the external sources are mapped to formats understood by the application and then used to answer the user's questions.	To fulfill this purpose, the CDS-System will use Direct hyperlinks and info buttons (see (Del Fiol et al. 2014)).
Optimizing process flow and workflow	The CDS-System shall use guidelines to guide the user's decisions and therapy pathway.	NCCN Guidelines
Monitoring of actions – guarding against errors	The CDS-System shall monitor changes performed by the user during patient-care to guard against errors.	Generate warnings and alerts upon detection of events and undesired conditions. These alerts are displayed in the GUI in a way that is unobtrusive to the user's workflow, to avoid alert fatigue.

Table 10 Principal purposes for CDS, and the key methodologies used (Greenes 2014a)

## Integration with the EHR and patient specificity

All derived CDS-System shall be integrated into EHR applications.

## RESTful API for CDS-System functionality

The functionality of all derived CDS-Systems shall be exposed to the EHR applications via RESTful APIs.

## Access management

Access to the CDS-Systems' APIs shall be managed, by leveraging authorization and authentication mechanisms. This is important since the CDS-Systems may access patient data, to provide recommendations.

## External CDS content

The CDS-Systems do not include a local knowledge-base. Consequently, the CDS-Systems shall use external information retrieval resources for providing CDS. The CDS-Systems can however be extended to include a knowledge base.

## Information model

The CDS-Systems provide an API that is independent of the information model used by invoking EHR applications. While this supports the portability of the CDS-Systems, it requires that application developers get acquainted to the CDS-System API.

### 7.2.2 Variability analysis

This section discusses how derived CDS-Systems may vary from each other, in other words how the product line shall support customization of the CDS-Systems. Variants annotated as “...” in the following symbolize support for extension.

#### VP1: EHR application services

As depicted in Figure 13, the integration of the CDS-Systems into the EHR shall be optional except for the Terminology Service which is used in selected free text fields of the EHR application.

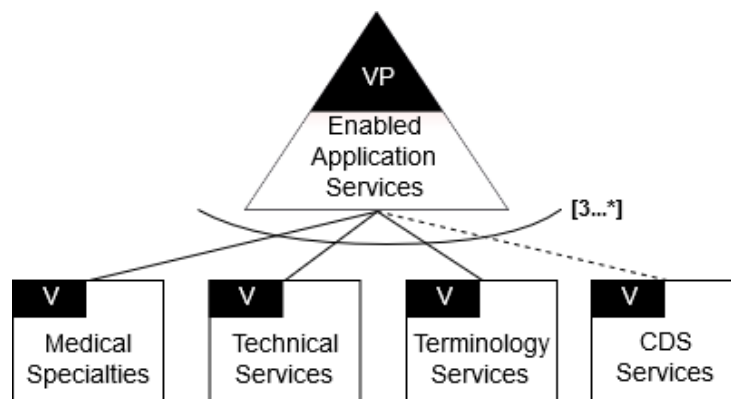


Figure 13. OVM enabled application services

#### VP2: Locus of control for information retrieval

Information for CDS is either retrieved proactively by the EHR application or on user demand. For example, the EHR application (i.e. the machine according to the terminology in the previous sentence) could be configured to invoke the CDS-System and retrieve desired information immediately after the user opens a patient's EHR. In this case, the CDS-System is perceived by the user to run in the background and offer recommendations proactively. Another approach could be to retrieve CDS information only after the user clicks an info button within the displayed EHR of a selected patient. This variation is to be implemented in client applications. The CDS-System backend shall support both approaches.

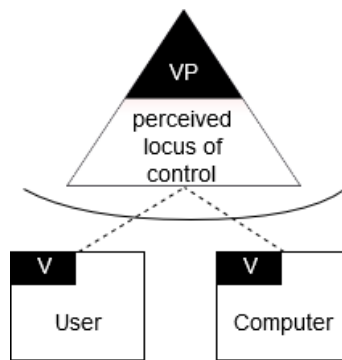


Figure 14 OVM perceived locus of control

### VP3: Supported medical specialties

The CDS-System shall provide decision support for several medical specialties. Figure 15 depicts the resulting variation point.

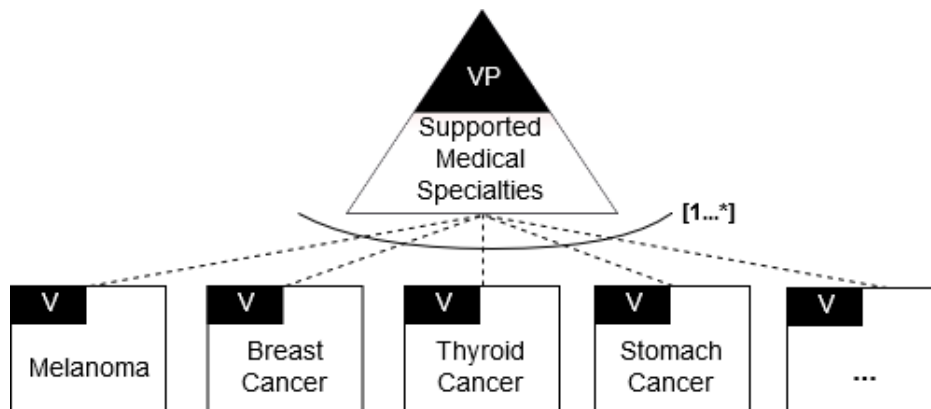


Figure 15 OVM supported medical specialties

### VP4: Enabled CDS-Services

The CDS-System shall provide CDS-Services for meeting the needs of physicians at the point of care. These include;

- Terminology service
- Drug interaction service
- Drug adverse events service
- Clinical trial service
- Evidence-based medical guideline service and a
- Literature (Medical publication) service

Derived CDS-Systems shall always provide a Terminology service and any combination of the above-mentioned medical information services. Figure 16, depicts the resulting variation point.

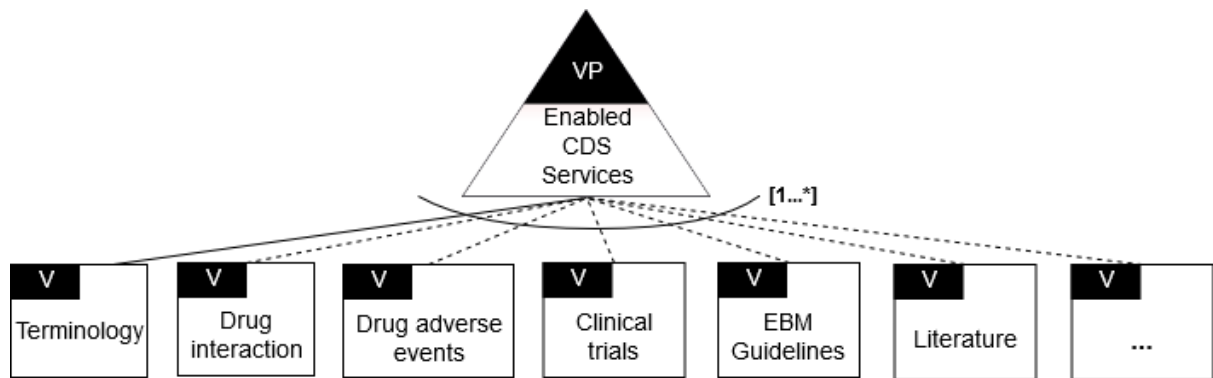


Figure 16 OVM enabled CDS-Services

### VP6: Displayed order of medical information services

The order in which the medical information services are displayed in EHR application GUIs, may vary depending on physicians' requirements.

### VP7: Information sources for medical information services

Information retrieval resources, may vary for any supported medical information services e.g. clinical trials could be retrieved from (WHO) or (ClinicalTrials.gov) and drug interaction could be retrieved from DrugBank (Wishart DS, Knox C, Guo AC, Shrivastava S, Hassanali M, Stothard P, Chang Z, Woolsey J.), ONCHigh (Phansalkar et al. 2012). Figure 17 depicts variability in the information sources for the drug interaction information service. At least one information retrieval resource shall be selected per enabled CDS.

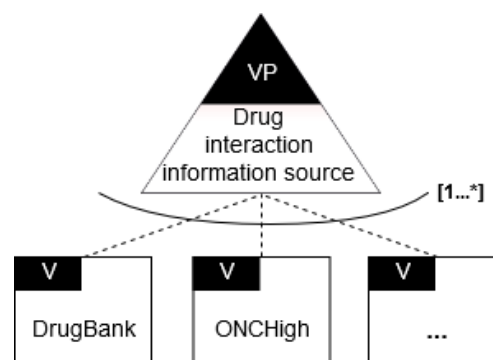


Figure 17 OVM drug interaction information sources

## 7.3 Domain design

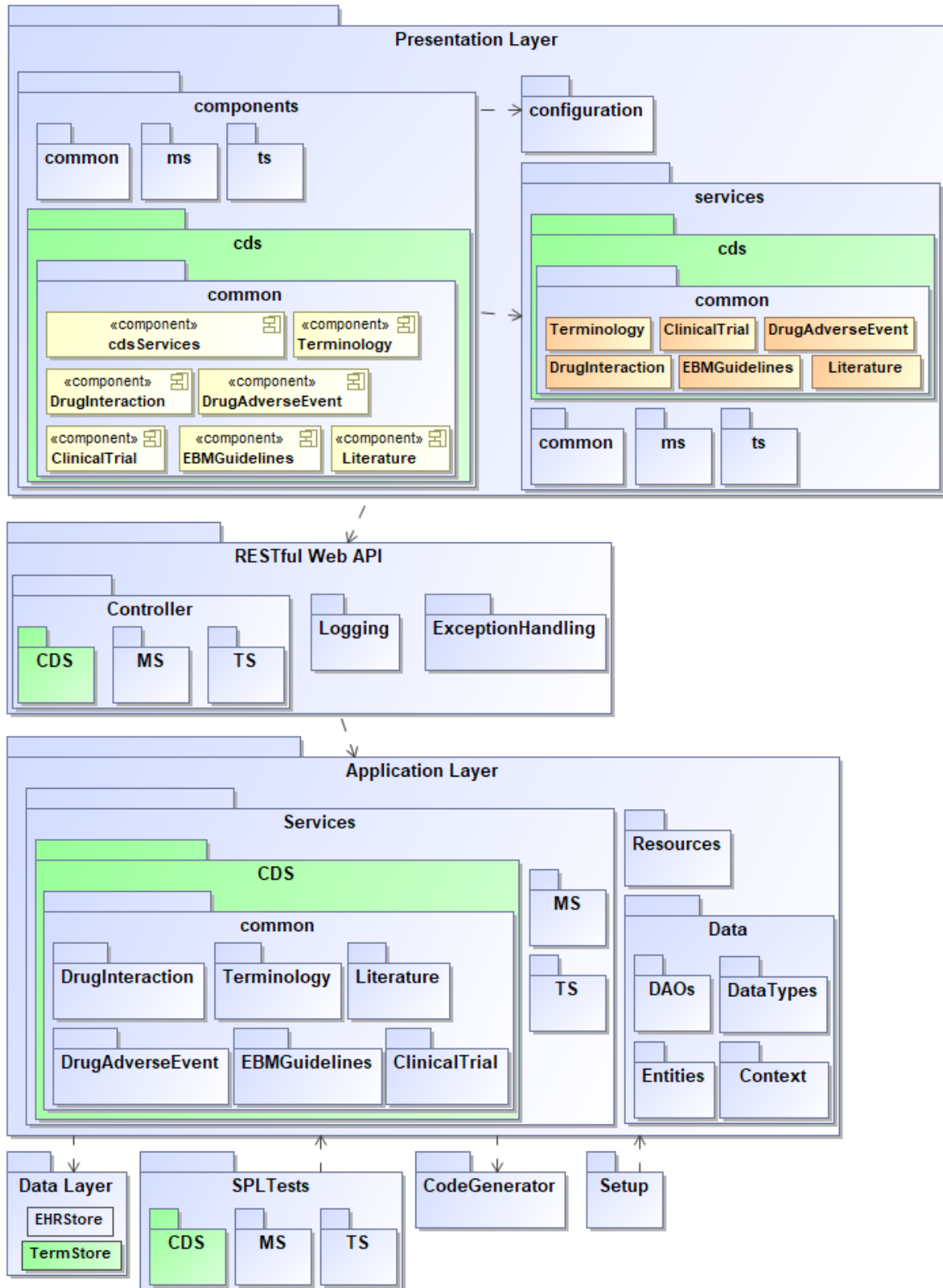


Figure 18 SAGE-CARE SPL Reference Architecture with CDS-Services

The goal of the domain design process is to develop the reference architecture for the product line based on identified common and variable requirements (see Chapter 7.2). This chapter

discusses the proposed reference architecture for the product line and presents reusable core assets that will support the realization of EHR applications with integrated CDS-Systems.

As depicted in Figure 18, the proposed reference architecture is organized into three Layers, namely the Presentation, Application and Data Layers. Layering provides for loose coupling, separation of concerns and accommodates changes to the architecture while assuring compatibility for existing products (Pohl et al. 2005). Packages highlighted in green contain CDS functionality. In the following, the layers are discussed in detail. The common, MS (Medical Specialty), CDS and TS (Technical Services) folders, are used in all layers to separate corresponding logic and enhance code maintainability.

- Common: Contains shared artefacts (i.e. components, services, classes, views etc.)
- MS: Contains artefacts implementing EHR functionalities or views
- CDS: Contains artefacts for providing CDS
- TS: Contains logic for technical services e.g. google places API, security (access management).

The Data Layer consists of two stores. The EHR Store for storing patient data, is implemented using a Microsoft SQL Database. Microsoft SQL Server Express 2016 or higher is required for leveraging row level security, as a multi-tenancy data storage model is adopted (see Figure 7), requiring tenant data isolation. The Term store is implemented using an Apache Solr server in version 5.1.0 (see Chapter 6.5). The term store is used to provide the terminology service, that is part of the CDS-Services discussed in Chapter 6.

The application layer implements the business logic of the product line applications in C#. The business logic is organized into services (small, simple, and reusable modules), resulting from the adoption of a SOA-based approach, as mentioned in Chapter 3.5. The Services package in the application layer contains technical services (TS), medical specialty (MS) services<sup>4</sup> and the CDS-Services discussed in Chapter 6 (see Figure 12). The Data package implements logic for accessing the EHR Store and uses Entity Framework as object-relational mapper (ORM). It is comprised of Data Access Objects that perform CRUD (Create, Read, Update, Delete) operations on the patient data in the EHR Store. Database configuration logic is implemented in the Context package e.g. logic for configuring row level security. The Resources package contains non-code files e.g. an excel spreadsheet used for code generation.

---

<sup>4</sup> MS Services are synonymous to EHR Services.

The RESTful Web API, facades the Application Layer. It provides APIs through which client applications access the services provided by the application layer (EHR and CDS-Services). the RESTful Web-API is implemented in C# using the ASP.NET Web-API framework.

The presentation layer is comprised of EHR and CDS view components that provide Graphical User Interfaces for user interaction. View components display data retrieved from the backend via services that implement RESTful client logic. Presentation layer applications (a.k.a. client applications) are accessed through a web browser, in the case of this thesis Mozilla Firefox. The presentation layer is implemented using HTML, CSS, Typescript, AngularJS, Bootstrap, AngularJS Material and jQuery.

The Setup package hosts logic for seeding the database with patient, tenant, administrator and user data. This is used for demonstration purposes.

The Code Generator contains logic for generating the Information Model using the aforementioned excel spreadsheet. SPLTests consists of unit tests for SPL core assets.

The domain design process must consider the results of the variability analysis and ensure that the reference architecture accommodates identified variabilities (Pohl et al. 2005, p. 221). The focus of this thesis being the CDS-Services, we in the following discuss identified CDS-related variation points, that must be implemented in the domain realization sub process.

### **Variability in the application layer**

The CDS-Services discussed in Chapter 6, are integrated into the Services/CDS package of the application layer as shown in Figure 18.

Using the integrated CDS-Services, application developers shall be able provide any of the following CDS-Services:

- Drug interaction
- Drug adverse events
- Clinical trial
- Literature
- Terminology: mandatory, used for autosuggestion in free text fields of EHR applications.
- Evidence-based medical guideline (new and SAGE-CARE specific)

This variation point is depicted in Figure 16. For each CDS-Service, application developers shall be able to select an information source. In this thesis, only the drug interaction service

provides more than one information source, namely, ONCHigh and DrugBank (see Figure 17). Finally, providing CDS in any developed application shall be optional (see Figure 13).

## Variability in the Restful Web API layer

Application developers shall implement controllers for calling enabled CDS-Services within the application layer.

## Variability in the Presentation layer

Domain design shall implement view components and services (client logic to access backend APIs) for all supported CDS-Services. The application developer shall select view components and services corresponding to the enabled CDS-Services in the application and RESTful Web API layers. The cdsServices component (in Figure 19) shall serve as orchestrating component for configuring supported CDS-Service (see Figure 19).

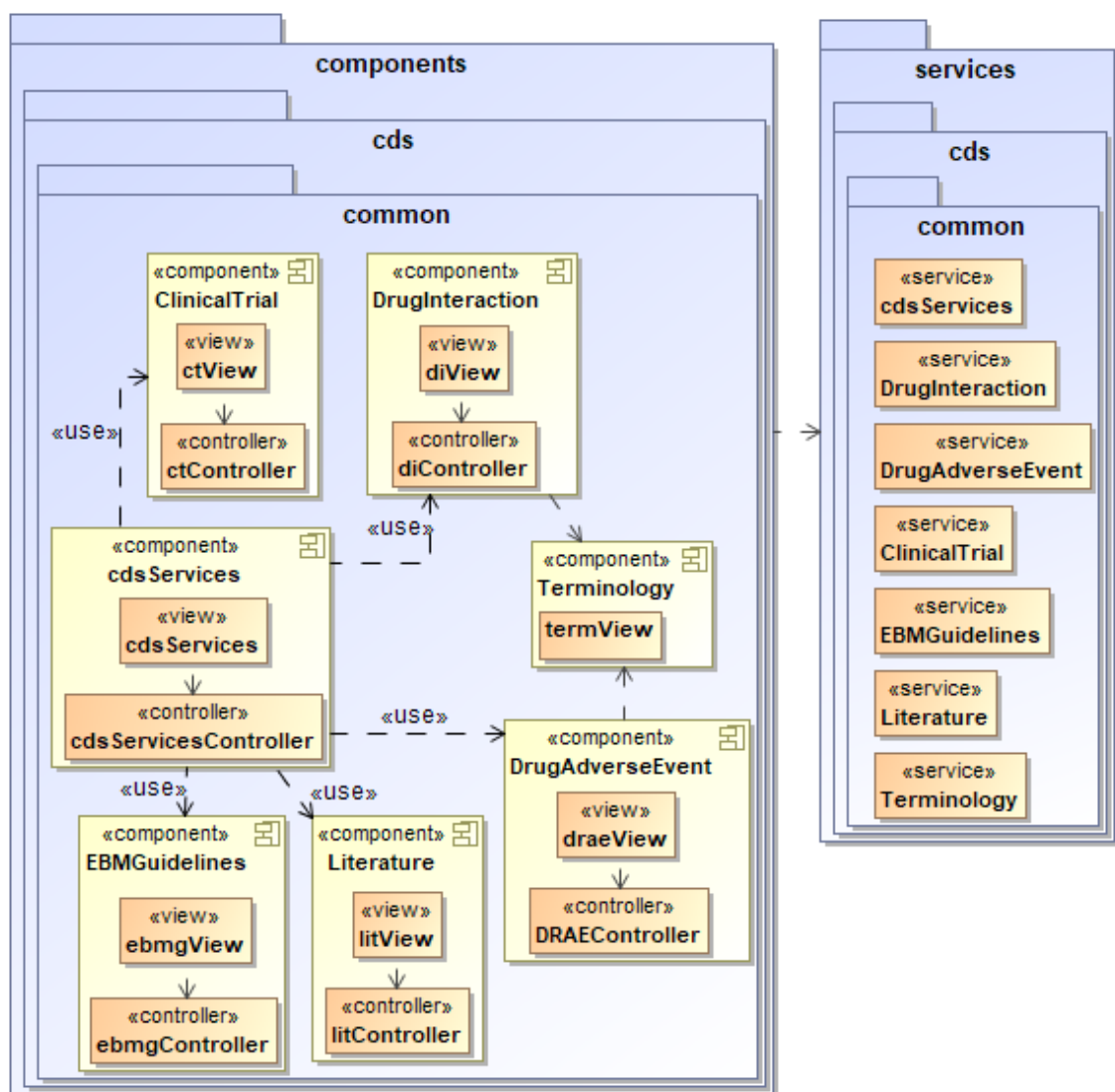


Figure 19 CDS core assets in SAGE-CARE Presentation Layer



## 7.4 Domain realization

The domain design process focuses on the design and implementation of core assets (components, interfaces etc.), based on the input from the domain design process i.e. the reference architecture and the selection of reusable software artefacts. The output of domain design includes, reusable components, interfaces and details on configuration support.

The outer packages of the SAGE-CARE SPL architecture (in Figure 18) are implemented as separate projects. Figure 20 shows the resulting projects.

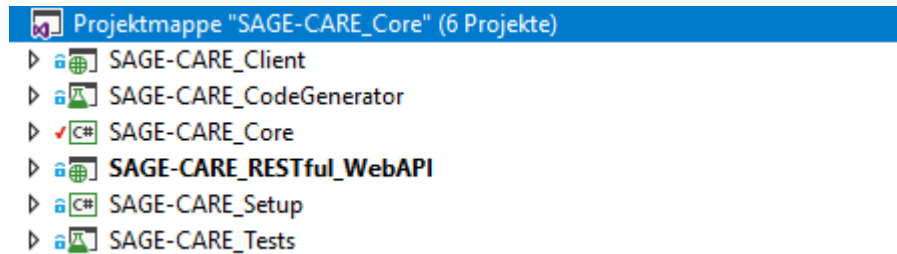


Figure 20 SAGE-CARE SPL projects

Inner packages of the SAGE-CARE SPL architecture are organized into folders within their corresponding projects. All projects except the SAGE-CARE\_Client project are implemented in C#.

The following sections discuss the developed core assets and configuration support for the SAGE-CARE\_Core, SAGE-CARE\_Client and SAGE-CARE\_RESTful\_WebAPI projects.

### 7.4.1 SAGE-CARE\_Core project

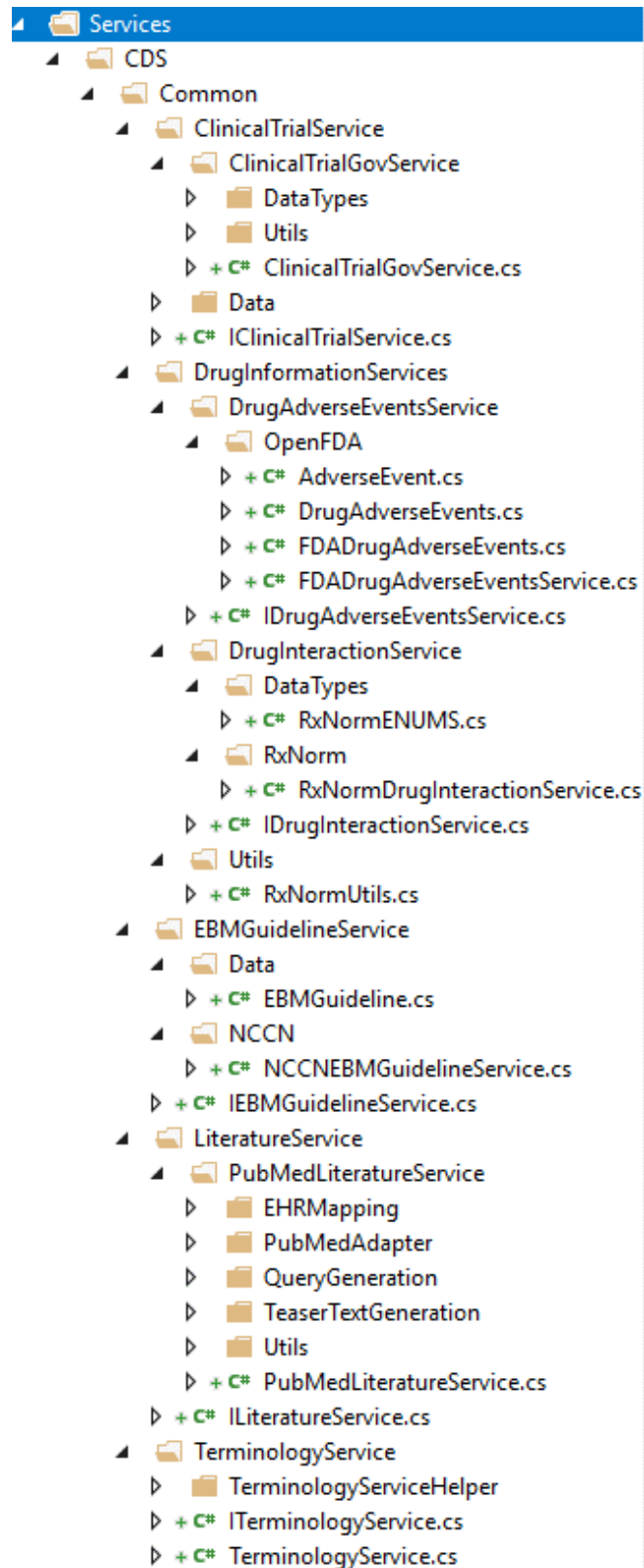


Figure 21 CDS Core Assets in SAGE-CARE\_Core project

Figure 21 shows an overview of available CDS core assets in the SAGE-CARE\_CORE project. Generally, each CDS-Service has a folder in *Services/CDS/Common*. The CDS-Service folders contain an Interface e.g. *IDrugInteractionService.cs*, and an implementation of the interface.

The implementation class is named according to the medical information retrieval resource from which CDS information is retrieved, e.g. `RXNormDrugInteractionService.cs`. The `Utils` folders contain utility classes (e.g. `RxNormUtils.cs`) that contain reusable code, to avoid code duplication in interface implementation classes.

To use a CDS-Service, application developers shall instantiate a selected implementation class and call the desired method as shown in Listing 17.

```
IDrugInteractionService rxnormDIService = new RxNormDrugInteractionService();
rxnormDIService.FindDrugInteractions(medications);
```

Listing 17 Using the drug interaction service

Table 11 provides an overview of available interfaces and their implementing classes.

Interface	Implementation
IClinicalTrialService	ClinicalTrialGovService
IDrugAdverseEventsService	FDADrugAdverseEventsService
IDrugInteractionService	RxNormDrugInteractionService
IEBMGuidelineService	NCCNEBMGuidelineService
ILiteratureService	PubMedLiteratureService
ILiteratureService	PubMedLiteratureService
ITerminologyService	TerminologyService

Table 11 Overview of available interface implementations for CDS-Services

## 7.4.2 SAGE-CARE\_RESTful\_WebAPI

No core assets are available in this project. However, application developers shall implement application specific controllers in the *Controllers/CDS/APPNAME* folder. The endpoints for implemented APIs shall be added to the *app/services/common/uriService/uriService.ts* file.

## 7.4.3 SAGE-CARE\_Client project

The client project is implemented using HTML, AngularJS, Typescript, CSS, Bootstrap, Angular Material.

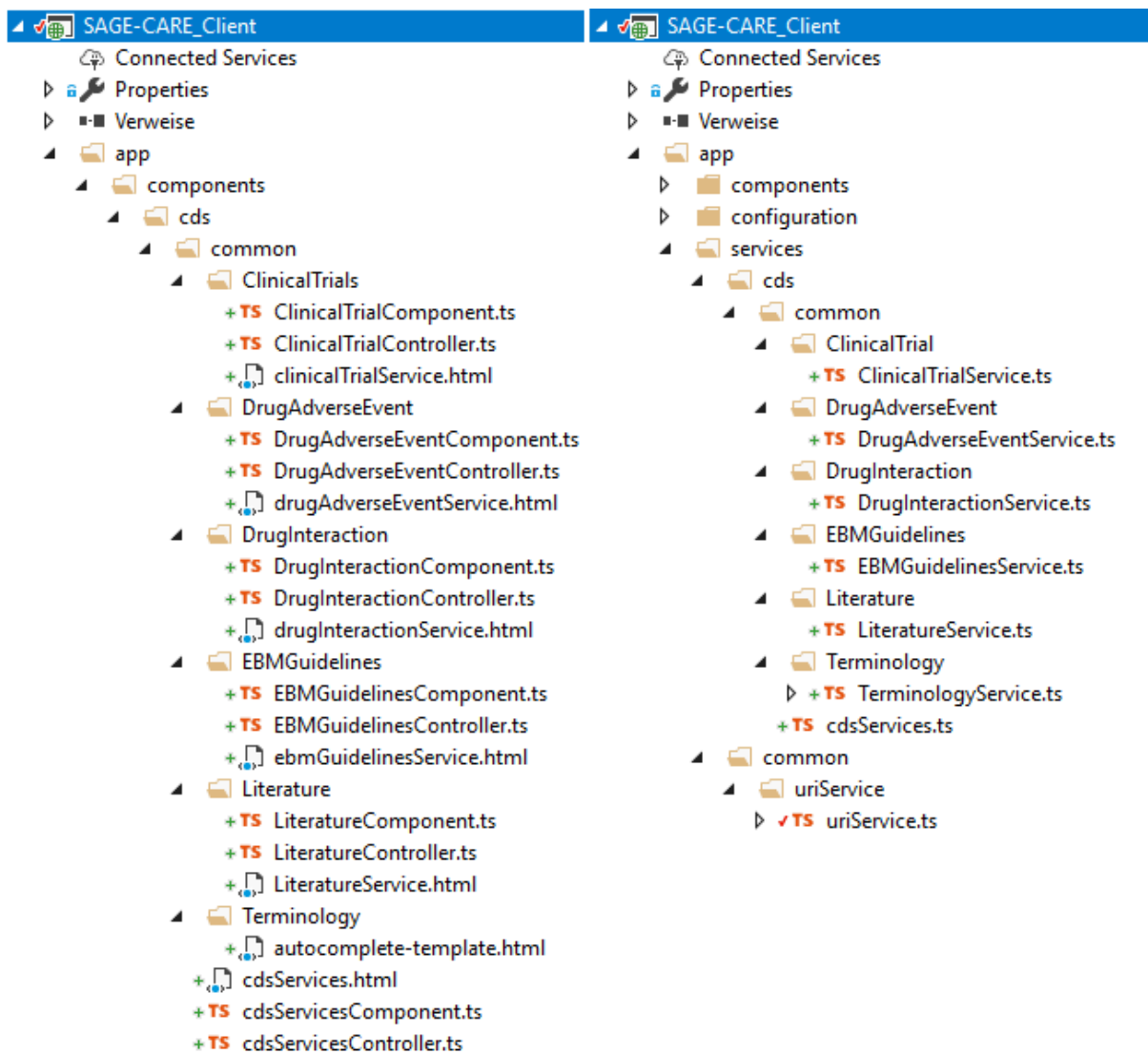


Figure 22 CDS Core Assets in SAGE-CARE\_Client project

Figure 22 shows CDS core assets within the SAGE-CARE\_Client project. The folder structure reflects the package ordering in Figure 18. Each CDS-Service has a component, controller, HTML template (view) and a service in *app/services/cds/common*. AngularJS data binding is used to synchronize the model (controller) and the view. All CDS-Services share a common flow of logic. Let's for example consider the drug interaction service. To retrieve data from the backend, the `DrugInteractionController` calls the `DrugInteractionService`. The `DrugInteractionService` retrieves the URL of the DrugInteraction API from the `uriService`. The `DrugInteractionService` then calls the DrugInteraction API within the `RESTful_WebAPI` project to retrieve drug interaction information from the backend. Retrieved information is then forwarded from the `DrugInteractionService` back to the `drugInteractionService.html`.

To include CDS-Services into any EHR application, application developers shall add the following line in the desired view file:

```
<div ng-include="'app/components/cds/common/cdsServices.html'" id="mis"></div>
```

#### Listing 18 One liner for displaying CDS-Services in EHR application

The included CDS-Systemervices.html, serves as a configuration file. It contains the list of CDS-Services the application developer aims to include in the EHR application. Application developers shall copy this file, rename it and make changes to include desired CDS-Services. The application developer shall then modify Listing 18, to point to the newly created file. The CDS-Systemervices.html is shown in Listing 22. This file implements variation points, VP1, VP4 and VP6, discussed in Chapter 7.2.2. It shall also be used for tenant-based activation of CDS-Services.

Application developers shall also modify the *app/services/common/uriService/uriService.ts*, to provide the URL of CDS-Services APIs in the RESTful\_WebAPI project.

The CDS-Services user interfaces are implemented as accordions (using Bootstrap) (see Figure 23). Initially all CDS-Services accordions are closed. A CDS-Service queries backend and external services only after the user opens the accordions. This approach is adopted so the CDS-Services remain unobtrusive to the clinical workflow (see Requirement 1 in Chapter 2).



Figure 23 CDS-Services user interaction interface

## Drug interaction service

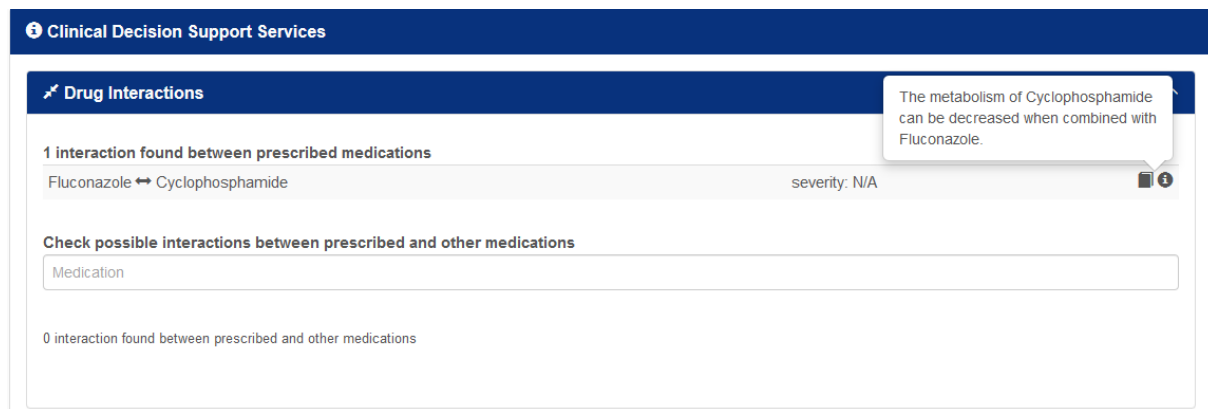


Figure 24 Drug interaction service user interface

When the user opens this accordion, the backend and corresponding external CDS webservices are queried to retrieve drug interaction information for prescribed drugs. The user can hover the info button on the right to get further information on the found interaction. Hovering the book icon reveals steps performed by the external services to find interactions e.g. the resolving of medication names (see Figure 25).



Figure 25 Additional information in drug interaction service user interface

Before prescribing a new medication, physicians can use the medication input field to check for interactions between the considered and prescribed medications. This way physicians can avoid adverse events due to interactions. Physicians are assisted in this process by the autocomplete service as shown in Figure 26. The results for interactions between prescribed and considered medications is displayed below the Medication input field as shown in Figure 27. The Medication field uses the ngTagsInput AngularJS library, that holds a list of tags e.g. the *ASPIRIN/CAFFEINE/PROPOXYPHENE* tag in Figure 27.

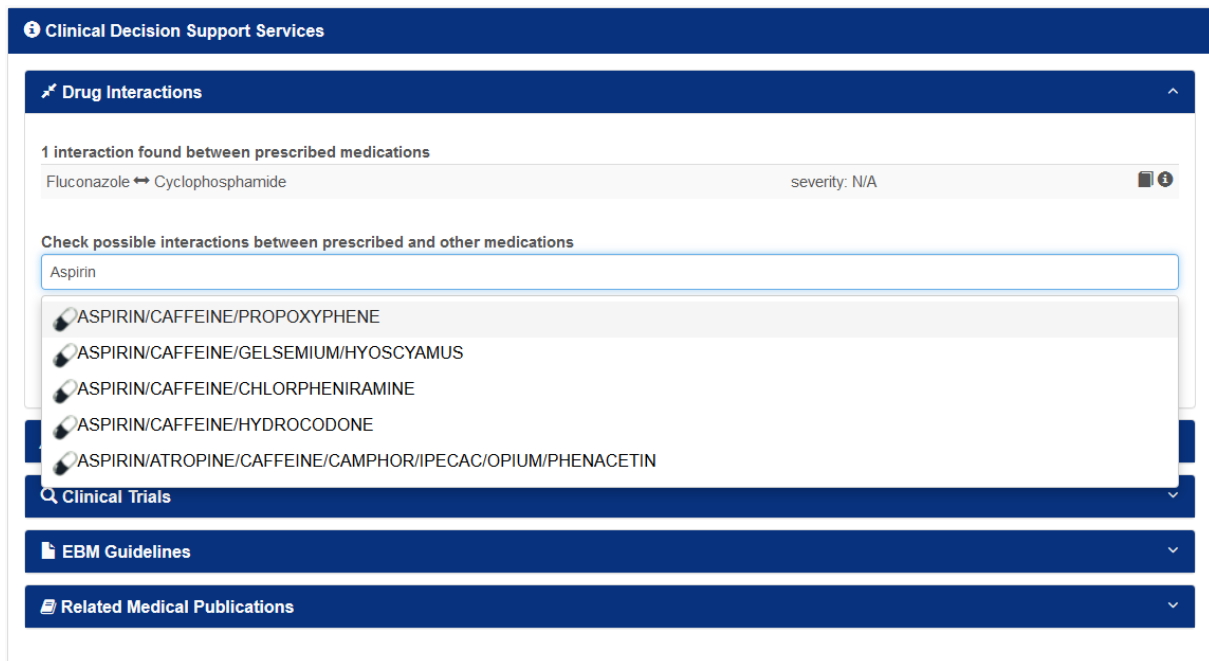


Figure 26 Autocomplete service in drug interaction user interface

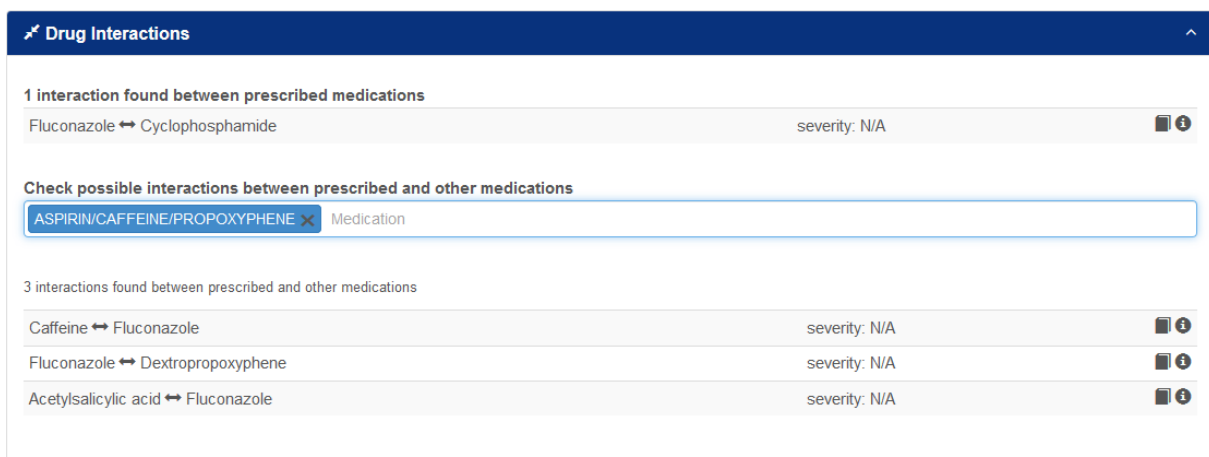


Figure 27 Entering Medications for interaction check in drug interaction service user interface

## Drug Adverse Events service

The drug adverse events service queries external CDS webservices after the accordion is opened. Adverse events are retrieved for prescribed as well as considered medications in the format: “Adverse event (Number of reports)”. The user interface also includes a Medication field with autosuggest functionality. Figure 28 and Figure 29 show the drug adverse event service user interface.

Drug Interactions

Drug Related Adverse Events

Possible drug-related adverse effects of prescribed medications

levothyroxin	No Information Available
bosentan 125 MG Oral Tablet	DEATH(7735), DYSPNOEA(4258), PNEUMONIA(2001), PULMONARY ARTERIAL HYPERTENSION(1787), CONDITION AGGRAVATED(1408),

Check possible drug-related adverse effects of other medications

warfa

Warfarin

Warfarin Sodium 10 MG Oral Tablet

Clinical Trials

Figure 28 Autosuggest service in drug adverse event service user interface

Drug Related Adverse Events

Possible drug-related adverse effects of prescribed medications

levothyroxin	No Information Available
bosentan 125 MG Oral Tablet	DEATH(7735), DYSPNOEA(4258), PNEUMONIA(2001), PULMONARY ARTERIAL HYPERTENSION(1787), CONDITION AGGRAVATED(1408),

Check possible drug-related adverse effects of other medications

Warfarin Medications

Warfarin No Information Available

Figure 29 Verifying drug adverse events of non-prescribed drugs

## Clinical trial service

The clinical trial service template displays a prepared link to the ClinicalTrials.gov webservice. Prepared here means that parameters with patient-specific data are added to the link. The user interface also displays the values of the parameters used in the link. Clicking on the link directs the physician to the result page at ClinicalTrials.gov (see Figure 31).

Clinical Trials

View clinical trials for: **Issue:** Thyroid Cancer | **Terms:** levothyroxin OR bosentan 125 MG Oral Tablet | **Gender:** Studies with FEMALE participants | **Age:** 30 years | **Age Group:** Adult | **Study Type:** All | **Study Phase:** All | **Recruitment status:** Recruiting | **Country:** Ireland | **City:** Cork | **Distance:** 0

Figure 30 Clinical trial service user interface

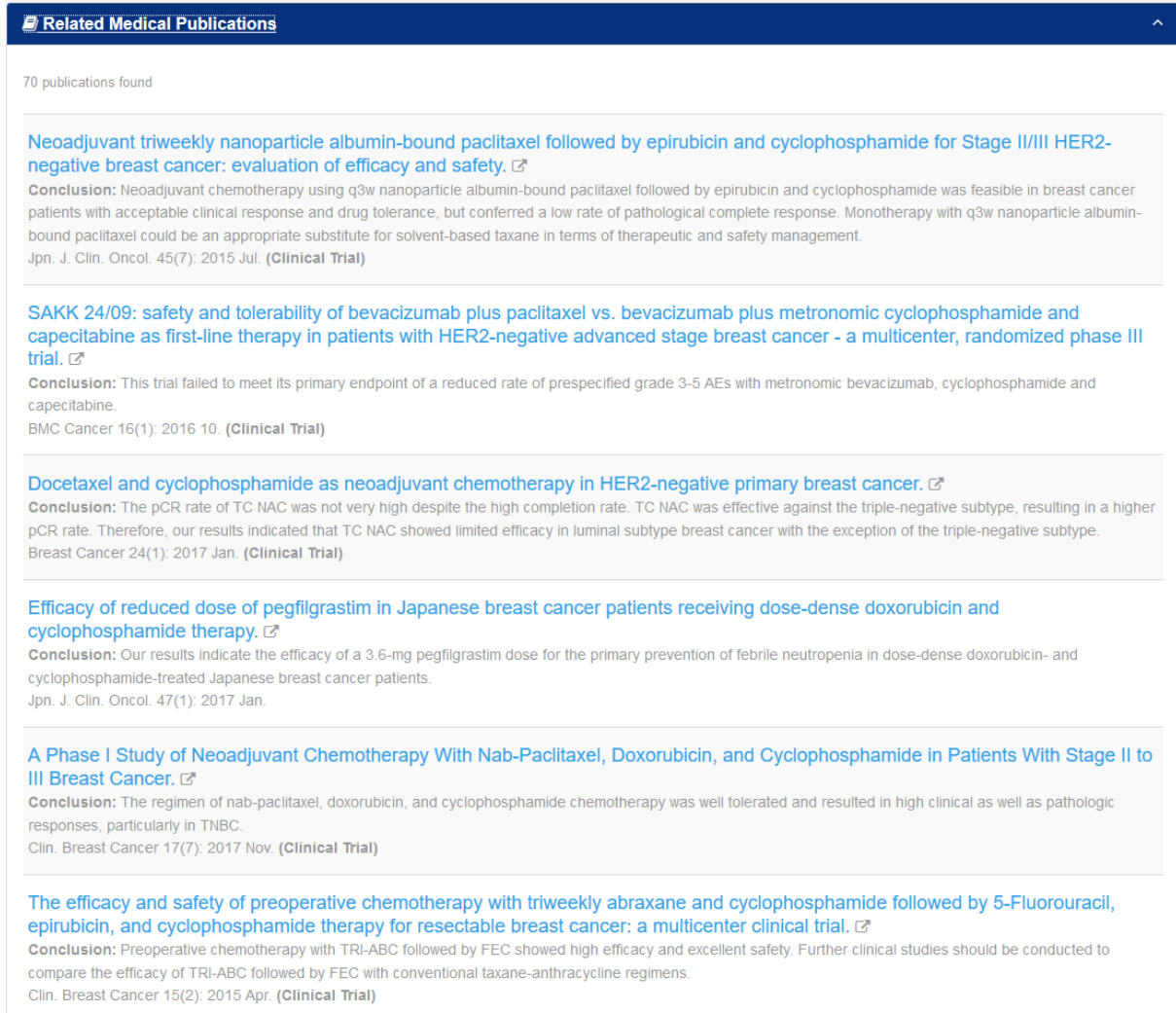


The EBM guidelines user interface uses an iframe to display NCCN Guideline PDF files. The displayed file is selected based on patient issues.

Figure 32 EBM Guidelines service user interface

## Literature Service

The literature service user interface simply displays a list of relevant medical publications together with a conclusion and a publication type. The conclusion is generated by leveraging natural language processing techniques to identify concluding phrases within the abstracts of found medical publications (Johannes Idelhauser 2016).



The screenshot displays the 'Related Medical Publications' section of a user interface. It features a dark blue header with the title and an upward arrow. Below the header, it states '70 publications found'. The main content area lists five publications, each with a title, a conclusion, and a citation. The titles are in blue and include a link icon. The conclusions are in black text, and the citations are in a smaller font at the bottom of each entry.

**Related Medical Publications**

70 publications found

**Neoadjuvant triweekly nanoparticle albumin-bound paclitaxel followed by epirubicin and cyclophosphamide for Stage II/III HER2-negative breast cancer: evaluation of efficacy and safety.** [↗](#)  
**Conclusion:** Neoadjuvant chemotherapy using q3w nanoparticle albumin-bound paclitaxel followed by epirubicin and cyclophosphamide was feasible in breast cancer patients with acceptable clinical response and drug tolerance, but conferred a low rate of pathological complete response. Monotherapy with q3w nanoparticle albumin-bound paclitaxel could be an appropriate substitute for solvent-based taxane in terms of therapeutic and safety management.  
Jpn. J. Clin. Oncol. 45(7): 2015 Jul. (Clinical Trial)

**SAKK 24/09: safety and tolerability of bevacizumab plus paclitaxel vs. bevacizumab plus metronomic cyclophosphamide and capecitabine as first-line therapy in patients with HER2-negative advanced stage breast cancer - a multicenter, randomized phase III trial.** [↗](#)  
**Conclusion:** This trial failed to meet its primary endpoint of a reduced rate of prespecified grade 3-5 AEs with metronomic bevacizumab, cyclophosphamide and capecitabine.  
BMC Cancer 16(1): 2016 10. (Clinical Trial)

**Docetaxel and cyclophosphamide as neoadjuvant chemotherapy in HER2-negative primary breast cancer.** [↗](#)  
**Conclusion:** The pCR rate of TC NAC was not very high despite the high completion rate. TC NAC was effective against the triple-negative subtype, resulting in a higher pCR rate. Therefore, our results indicated that TC NAC showed limited efficacy in luminal subtype breast cancer with the exception of the triple-negative subtype.  
Breast Cancer 24(1): 2017 Jan. (Clinical Trial)

**Efficacy of reduced dose of pegfilgrastim in Japanese breast cancer patients receiving dose-dense doxorubicin and cyclophosphamide therapy.** [↗](#)  
**Conclusion:** Our results indicate the efficacy of a 3.6-mg pegfilgrastim dose for the primary prevention of febrile neutropenia in dose-dense doxorubicin- and cyclophosphamide-treated Japanese breast cancer patients.  
Jpn. J. Clin. Oncol. 47(1): 2017 Jan.

**A Phase I Study of Neoadjuvant Chemotherapy With Nab-Paclitaxel, Doxorubicin, and Cyclophosphamide in Patients With Stage II to III Breast Cancer.** [↗](#)  
**Conclusion:** The regimen of nab-paclitaxel, doxorubicin, and cyclophosphamide chemotherapy was well tolerated and resulted in high clinical as well as pathologic responses, particularly in TNBC.  
Clin. Breast Cancer 17(7): 2017 Nov. (Clinical Trial)

**The efficacy and safety of preoperative chemotherapy with triweekly abraxane and cyclophosphamide followed by 5-Fluorouracil, epirubicin, and cyclophosphamide therapy for resectable breast cancer: a multicenter clinical trial.** [↗](#)  
**Conclusion:** Preoperative chemotherapy with TRI-ABC followed by FEC showed high efficacy and excellent safety. Further clinical studies should be conducted to compare the efficacy of TRI-ABC followed by FEC with conventional taxane-anthracycline regimens.  
Clin. Breast Cancer 15(2): 2015 Apr. (Clinical Trial)

Figure 33 Literature service user interface

## Terminology Service

The template is used by the drug interaction and drug adverse event service templates for autosuggestion in their respective Medication fields as shown in Figure 26 and Figure 28. The Terminology service template is also used in fields of the EHR e.g. for entering medication prescriptions (see Figure 34).

**Patient**

MRN: 75s348vf

**Personal Data**

**Sample Issue**

Name: Thyroid Cancer

Description: A Sample Thyroid Cancer Issue for Demonstration purposes

Date: 02/04/2017

Float: 1,7

Stage: IIB

Bool: true false

**Medication**

**Medication**

**Medication**

Drug Name: warfar

Warfarin

Warfarin Sodium 10 MG Oral Tablet

Save

Figure 34 Terminology service autosuggest in EHR medication field

## 7.5 Domain Testing using Sample Application Strategy

This section aims to evaluate domain artefacts from domain requirements engineering, domain design and domain realization processes. The evaluation is performed using the Sample Application Strategy. This section therefore discusses the integration of CDS-Services in a sample EHR application and aims to address the following questions:

- What shall application developers do to integrate CDS in EHR applications?
- How easy can CDS-Services be integrated in EHR application?

### 7.5.1 Overview of the sample EHR application

The sample EHR application manages data for patients suffering from a sample issue. The sample application data model is shown in Figure 42. Figure 35 shows an example of a Breast Cancer SampleIssue linked to a patient (whose Medical Record Number (MRN) is 75s4X), person, consultant and an organization via HL7Relationship associations.

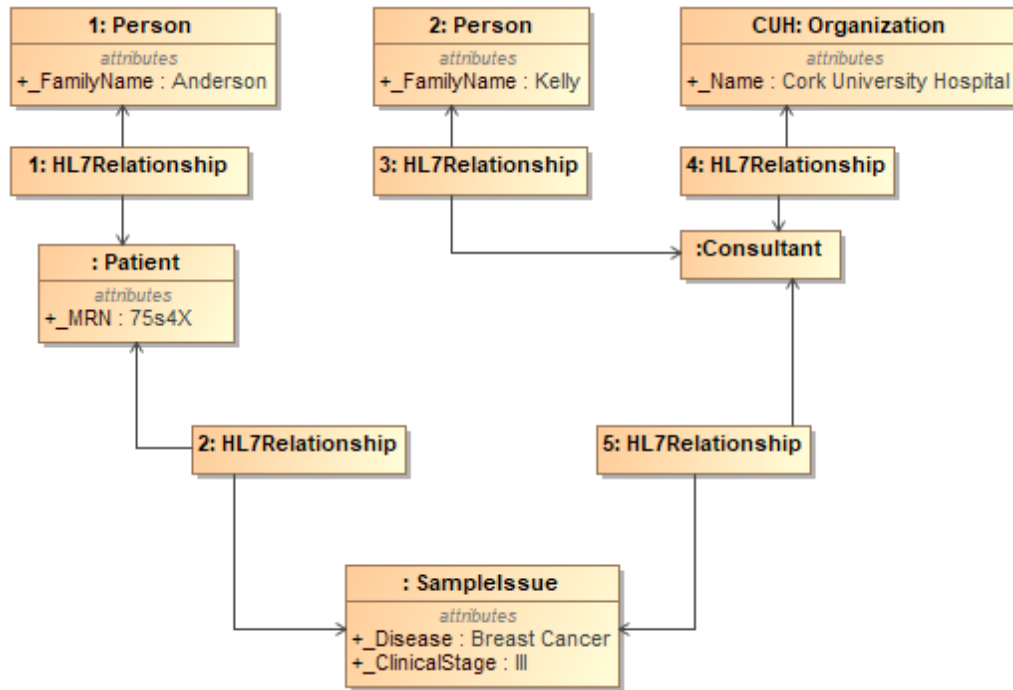


Figure 35 Breast cancer SampleIssue

A folder called Sample is created at different points within the folder structure of the different projects to host code specific to the sample application (see Figure 36). The SampleIssueDao is used to perform sample application specific CRUD operations.

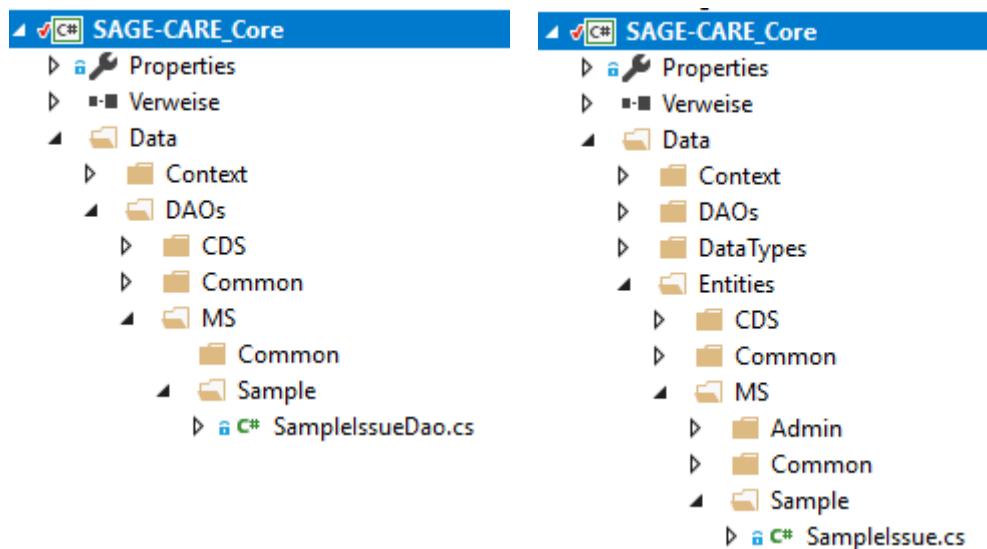


Figure 36 Introducing the Sample folder for Sample application classes

The following sections discuss the integration of CDS-Services in the sample application.

## 7.5.2 Core project

To integrate CDS-Services, the application developer creates a Sample folder in Services/CDS. The application developer then creates Sample CDS-Service classes in the Sample folder e.g. SamleDrugInteractionService (see Figure 37).

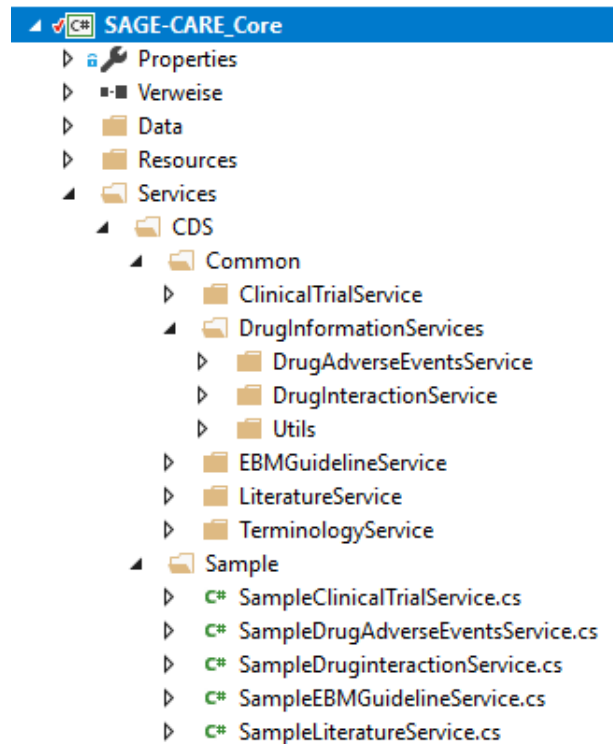


Figure 37 CDS-Service classes for Sample application in SAGE-CARE core project

```

17 public class SampleDrugInteractionService
18 {
19     private static readonly IDrugInteractionService rxnormDIService = new RxNormDrugInteractionService();
20
21     1-Verweis | Daniel Ebanja, vor 55 Tagen | 1 Autor, 1 Änderung
22     public static IList<DrugInteraction> GetDrugInteractions
23         (int patientId, IList<string> otherMedNames)
24     {
25         var patient = (Patient)HL7ObjectDao.FindById(patientId);
26         if (patient.Equals(null) || patient == null)
27         {
28             throw new ArgumentException();
29         }
30
31         var prescribedMedications = SampleIssueDao.GetPatientMedications(patient.Id)
32             .Where(m => m.Status == MedicationStatus.ACTIVE).ToList();
33         if (prescribedMedications.Count == 0) return null;
34
35         var prescribedMedicationLabels = prescribedMedications.Select(m => m.Label).ToList();
36
37         return rxnormDIService.FindDrugInteractions(otherMedNames.Union(prescribedMedicationLabels).ToList());
38     }

```

#### Listing 19 Using the Sample drug interaction CDS-Service - SampleDrugInteractionService

Listing 19 shows the implementation of the SampleDrugInteractionService. All sample CDS-Service classes have a common flow of logic. Patient-specific data is retrieved from the EHR store (see Lines 24 and 30), processed (Line 34) and forwarded to the CDS-Service (Line 36).

### 7.5.3 RESTful\_WebAPI

The application developer creates a Sample folder in Controllers/CDS and adds an ASP.NET Web API controller (inherits from ApiController) class to the folder, namely, SampleCDS-SystemservicesController (see Figure 38)

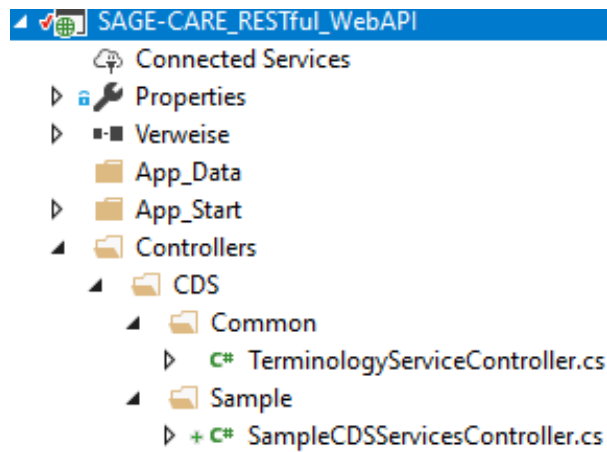


Figure 38 Creating the Sample application controller

```
public class SampleCDSServicesController : ApiController
{
    [Route("clinicalTrials/{patientId}/")]
    [HttpGet]
    public ICollection<ClinicalTrials> GetClinicalTrials(int patientId)
    {
        return SampleClinicalTrialService.FindClinicalTrials(patientId,
            SampleIssueDao.GetPersonByPatientId(patientId).Id);
    }

    [Route("drugInteractions/{patientId}/")]
    [HttpGet]
    public IList<DrugInteraction> GetDrugInteractions(int patientId,
        [FromUri] List<string> otherMedications)
    {
        return SampleDruginteractionService
            .GetDrugInteractions(patientId, otherMedications);
    }

    [Route("getDrugAdverseEvents/{medication}/")]
    [HttpGet]
    public ICollection<DrugAdverseEvents> GetDrugAdverseEvents(string medication)
    {
        return SampleDrugAdverseEventsService.GetDrugAdverseEvents(medication);
    }

    [Route("patientDrugAdverseEvents/{patientId}/")]
    [HttpGet]
    public ICollection<DrugAdverseEvents> GetDrugAdverseEvents(int patientId)
    {
        return SampleDrugAdverseEventsService.GetDrugAdverseEvents(patientId);
    }

    [Route("medicalPublications/{issueId}/")]
    [HttpGet]
    public DocumentsAndEhrTerms SearchLiteratureForIssueId(int issueId)
    {
        var issue = HL7ObjectDao.FindById(issueId) as SampleIssue;
        return new SampleLiteratureService().SearchLiterature(issue);
    }
    ...
}
```

Listing 20 Specifying the Sample application Web API in SampleCDSServicesController.cs

Listing 20 shows the implementation of the `SampleCDSServicesController` class. The `SampleCDSServicesController` specifies the Web API for the sample application business logic in the Core project and handles incoming HTTP/S requests by forwarding them to corresponding sample CDS-Services.

### 7.5.4 Client project

The application developer creates a sample folder in `app/components/cds/sample`. The application developer then adds the `SampleCDSServices.html` file to this folder, copies the code from the `cdsServices.html` file and configures enabled CDS-Services (see Figure 39).

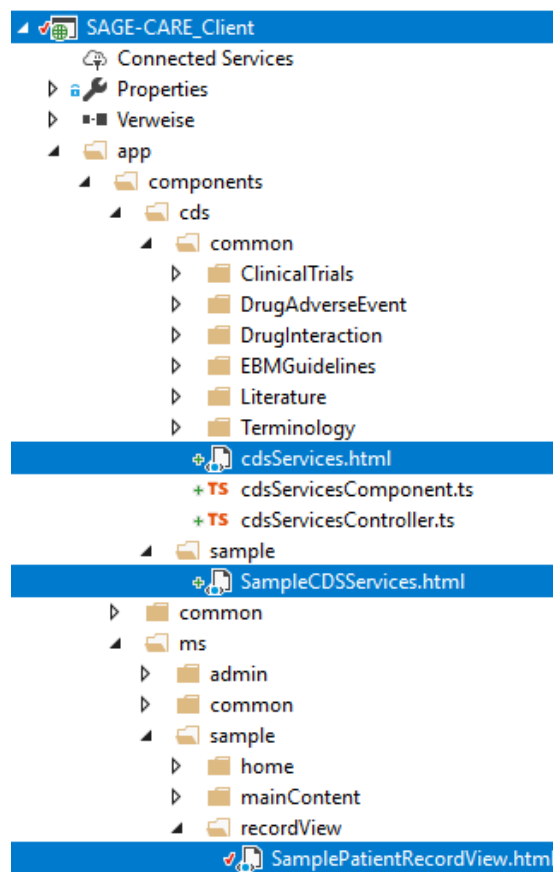


Figure 39 Sample application core assets in the client project

To display the CDS-Services in the GUI of the sample EHR application, the application developer adds just a line of code to the `SamplePatientRecordView.html` file as shown in Listing 21.

```

<div class="container">
  <sample-relationship-sidebar parent-ctrl="$ctrl"></sample-relationship-sidebar>
  <div class="root-composite">
    <div class="root-composite-header">...</div>
    <div class="root-composite-body">
      <record ng-if="$ctrl.getModel()" abstract-entity-component="$ctrl.getModel()" parent-ctrl="$ctrl" component-type="$ctrl.getDetailComponentType()">
      </record>
      <div class="row">
        <div class="col-md-12">
          <div class="col-md-9">
          </div>
          <div class="col-md-3">
            <button type="submit" class="btn btn-primary btn-lg btn-block" ng-click="$ctrl.updateData(); $ctrl.fetchHistory()" translate="SAVE"></button>
          </div>
        </div>
      </div>
    </div>
  </div>
  <br />
  <div ng-include="app/components/cds/sample/SampleCDSServices.html" id="cds"></div>
  <div class="statusbar-expanded navbar-fixed-bottom">...</div>
  <div class="statusbar navbar-fixed-bottom">...</div>
</div>

```

Listing 21 SamplePatientRecordView.html - introducing CDS-Services UIs in the sample EHR application



## 8 Evaluation

The aim of this section is to evaluate the, in this thesis, proposed and prototyped concepts. Evaluation is performed based on the requirements in chapter 2 and guidelines from the Guides Project Checklist tool (BVBA 2017a, 2017b).

In the following, each requirement is evaluated separately:

The CDS-Services shall

1. Be integrated into the clinical workflow:

This requirement is obviously met. The proposed approach for providing CDS, is passive i.e. CDS-Services are queried only on user demand. Furthermore, the CDS-Services UIs are situated beneath the EHR UIs to enforce unobtrusiveness to the physician's workflow. The CDS-Services also neither generate alerts nor return recommendations that require action from the physician.

What is the impact of adding CDS-Services to the existing EHR application? (BVBA 2017b)

Conceptually, the CDS-Services are only used on user demand. The EHR application is therefore not impacted by the introduction of the CDS-Services. However, the response times of the CDS-Services are longer than those of the EHR application, as the CDS-Services query external CDS information retrieval resources.

Can the CDS-Services improve the workload or work processes? (BVBA 2017b)

At least an effort is made by the EBM guidelines service. A proper assessment requires user feedback.

2. Provide relevant information:

Does the decision support contain accurate information that is pertinent to the care of the patient? (BVBA 2017b)

The CDS-Services provide patient-specific recommendations. This is possible due to the integration of the CDS-Services in the EHR. To retrieve CDS content the EHR application invokes the CDS passing patient-specific information.

Assessing the accuracy of provided recommendations is not obvious as it requires feedback from the intended users. However, the selected information retrieval resources are widely used in the health care domain.

Do the CDS-Services address the information needs of the physicians? (BVBA 2017b)

In chapter 4, the information needs of physicians at the point of care were identified to be mainly associated with diagnosing health conditions, finding best treatment methods (therapies) and drug prescriptions. Provided CDS-Services aim to address these needs except for providing information for diagnosing health conditions.

Table 3 presents provided CDS-Services and addressed physician's information needs.

Is it clear to the users why the decision support information is provided for a given patient?(BVBA 2017b)

The UIs of the CDS-Services present patient-specific data that is used for querying CDS-Services. However, the assessment of this requirement should be complemented with user feedback.

3. Provide latest information:

Table 5 provides a list of information retrieval resources used by the CDS-Services and how frequent the information is updated. However, a proper assessment also requires feedback from physicians.

4. Pro-actively search and provide decision support without requiring entry of already existing patient data:

This requirement is met thanks to the integration of the CDS-System in the EHR. Physicians are not required to enter data that already exist within a patient's EHR.

5. Be intuitive and easy to use:

Assessing this requirement is not obvious as it requires feedback from users. This demands that a usability test must be performed which is out of the scope of this thesis. This is considered as a point for future work.

6. Be comprehensive:

Like the previous requirement, this requirement is also not obvious to assess. This depends on the selected information retrieval resources and requires feedback from users. Regarding the information retrieval resources, effort was performed to select widely used information retrieval resources. However, the use of commercial information retrieval resources may be a major step in this direction.

7. Have low response times:

This requirement is not always met. The response times vary per CDS-Service. Also, it is important to consider that patient-data is first retrieved from the EHR store before queries are placed to the information retrieval resources.

Furthermore, the product line architecture shall

8. Support easy integration of CDS-Services into EHR applications:

This requirement is met. The steps required to integrate CDS-Services into EHR applications are configuration steps and do not demand much effort (see chapter 7.5). The steps could be summarized as, copying configuration files, editing these (commenting undesired sections), and finally including one line of code (LOC) at a desired point in an HTML file.

9. Enable introducing new CDS-Services, with moderate implementation effort:

This requirement is met: The proposed concept provides a pattern for the introduction of a new CDS-Services. The application developer shall specify a properly generalized interface for the new CDS-Service and implement a RESTful client logic in classes that realize the interface.

10. Enable introducing new information retrieval resources, with moderate implementation effort.

To introduce a new information retrieval resource for an existing CDS-Service, developers shall create a class that realizes the interface specified for that service and provide RESTful client logic for the targeted information retrieval resource. This requirement is therefore met.

## 9 Related Work

SOA has been proposed as a promising approach for developing health information systems that support interoperability. Many initiatives have been and are performed to provide standard-based SOA resources. Some of these initiatives include:

- Healthcare Services Specification Project

Healthcare Services Specification Project (HSSP) is a collaborative effort of HL7 and the Object Management Group (OMG). The aim of HSSP is to provide standards for SOA-based services.

- OpenCDS:

OpenCDS is a “collaborative effort to develop open-source, standards-based clinical decision support (CDS) tools.”(OpenCDS)

OpenCDS provides a platform that supports scalability and interoperability of Decision Support Services (DSS). OpenCDS supports clinical decision support standards like HL7 Decision Support Service (HL7 DSS), HL7 Virtual Medical Record (HL7 vMR) and HL7 FHIR (HL7 FHIR 2017). The OpenCDS has been widely adopted (Welch et al. 2014b; ICE - CDS Framework Wiki; Welch et al. 2014a)

- Healthcare Services Platform Consortium

Health Services Platform Consortium (HSPC) is a collaborative effort that aims to provide a SOA-based “healthcare services platform” that supports development of high quality and interoperable health care applications at reduced costs. HSPC supports the following standards:

Data exchange standards	HL7 FHIR
Terminology standards	SNOMED CT(SNOMED International), LOINC(LOINC), RxNorm(NLM 2018)
EHR integration	SMART(SMART Health IT)

- SMART

“SMART Health IT is an open, standards based technology platform” that facilitates the development of interoperable applications. SMART provides “open standards, open source tools” and apps. (SMART Health IT)

# 10 Conclusion and Future Work

## 10.1 Conclusion

In this thesis, reusable CDS-Services were proposed to facilitate the integration of CDS in EHR applications. The proposed concept aimed to fulfill both the requirements of the product line owner and those of intended users, in this case physicians.

To achieve the thesis goals, the information needs of physicians at the point of care (decision making time) were identified (see chapter 4). Studies reveal that the information needs of physicians at the point of care are related to diagnosing patient health issues, identifying best treatment methods and drug prescriptions. To meet their information needs, studies reveal that physicians mainly refer to internet-based health information retrieval resources.

In this thesis, the proposed CDS-Services therefore implement RESTful client logic to retrieve CDS information from such medical information retrieval resources that are freely accessible and provide an API. Implemented CDS-Services include a drug interaction service, a drug adverse events service, a clinical trials service, an EBM guideline service and a literature service.

Several architectural approaches for developing CDS-Systems were identified and evaluated. Studies reveal a wide adoption of SOA-based approaches for developing CDS-Systems. Providing CDS functionality as a web service enables access to several EHR applications and other health information systems e.g. CPOE systems. The SOA approach thus suits the reusability requirements of product lines. The proposed concept therefore adopts a SOA-based approach and places APIs in front the CDS-Services, Terminology Services and EHR Data Access Services (see Chapter 6). In Chapter 7, the proposed concept is applied to the SAGE-CARE product line of EHR applications. Core asset development activities are discussed based on the SPLE Framework of (Pohl et al. 2005). The feasibility of the proposed concept and the developed CDS core assets is asserted by prototyping a sample EHR application with integrated CDS-Services.

The evaluation in chapter 8 reveals that the requirements for the thesis were met, apart from a few that require feedback from users. Getting user feedback requires that a usability study must be performed, but this is out of the scope of this work and considered as a point for future work.

## 10.2Future Work

It is intended that the sage care product line presented in this thesis will be promoted to a commercial product, that will be marketed by the NSilico Lifesciences Ltd. company. Future work is required for this purpose.

In order to asses requirements 5, 6 and the general usability of the CDS-Services, a usability study should be performed with physicians (Johannes Idelhauser 2016).

Furthermore, the use of commercial information retrieval resources is advised. This will require the use of standardized CDS APIs. Amongst the many initiatives performed to provide standardized SOA-based CDS, the HL7 DSS is advised, due to its wide adoption. In case the HL7 DSS is adapted, the developed CDS core assets (for the application, RESTful API and presentation layers) could be used to implement a CDS webservice as in the case of SEBASTIAN (Kawamoto and Lobach 2005).

Another consideration for future work is the implementation of further CDS-Services e.g.:

- Drug information service: This service shall mainly provide information available in medication leaflets, to enhance patient education (Johannes Idelhauser 2016). The DailyMed (DailyMed 2018) service offers a freely accessible API that can support this functionality
- Other proposed CDS-Services include, drug-dosing, drug-allergy-, drug-food-, drug-disease-, and drug-pregnancy-interaction services (Kuperman et al. 2007)
- Differential diagnosis service: This service shall retrieve patients symptoms and provide diagnostic decision support by proposing a list of diagnoses to be considered. (Hoffer et al. 2005; Henderson and Rubin 2013; Bond et al. 2012)

Different physicians may require switching between information retrieval sources for a given CDS-Service. To provide this functionality, variation mechanisms should be implemented, that support late binding of supported information retrieval resources (Gacek and Anastasopoulos 2001).

# 11 Appendices

## 11.1 Appendix A

### Drug Information retrieval resources

Name	Descriptions	API	Access	Drug Information	Drug Interactions	Adverse Events	Drug Announcements/Recalls
DailyMed	Website by U.S. National Library of Medicine (NLM), provides high quality and up-to-date drug labels. Updated daily by FDA. Documents use structured XML format	yes	public & free	✓	✓	✓	
DrugBank	Database with pharmacological drug information on drugs and their targets. Longer update periods. Links to DrugBank for nearly all drugs on Wikipedia. Drug interactions feature no information on their severity.	yes	public & free		✓		
Drugs.com	Website with drug information, pill identification and drug interaction checker for patients and for health professionals. Also provides data on modified drug labels. Prohibited to incorporate into any kind of IR system.	no	public & free	✓	✓	✓	
Electronic Medicines Compendium	Information on drugs licensed for use in the UK. Contains Summaries of Product Characteristics and Patient Information Leaflets	no	public & free	✓	✓	✓	
Expocrates	Point-of-care medical information about drugs, diseases and diagnostic tools over website or mobile app also features news feed of product announcements and medical news.	no	partly free / subscription needed.	✓	✓	✓	✓
MedlinePlus Connect	Service by NLM, provides unstructured natural language	yes**	public & free	✓ *		✓ *	

	drug information/labelling and health topic overviews						
Medscape	Many clinical information resources available over website or mobile app. Articles updated yearly.	no	free, registration required	✓	✓	✓	
OpenFDA API	Public API on reported Adverse Effects, drug labelling and drug recall reports. Data consists of individual reports and must be aggregated in order to use it.	yes	public & free			✓	✓
ResearchAE	Adverse effects experimental research application based on OpenFDA data. Not to be used in clinical settings.	no	public & free			✓	✓
RxNav	Provides access to different drug resources like RxNorm, NDF-RT and DrugBank. Drug normalisation over different codes and systems by using RxNorm, drug interactions from DrugBank.	yes	public & free		✓		
SIDER	Aggregated data on side effects for drugs target prediction from publicly available sources. Infrequent updates. Download of dataset possible.	no	public & free			✓	
Wolters Kluwer Clinical Drug Information	Commercial drug information APIs including interaction, adverse effects, indications and mapping to RxNorm.	yes	commercial	✓	✓	✓	

Table 12 Drug Information Data Sources (Johannes Idelhauser 2016)



## 11.2 Appendix B

### Clinical Trial information retrieval resources

Name	Description	API	Access	Type	Country
ClinicalTrials.gov	Trial registry from US National Institute of Health. 39% are U.S. only trials.	yes	Public & free	Search Service	Worldwide with a focus on U.S. (39%)
EU Clinical Trials Register	Clinical Trials Register for trials in the EU	no	Public & free	Register	European Union
German Clinical Trials Register	German clinical trial register. Also imports trials from clinicaltrials.gov that are located in Germany	no	Public & free	Register	Germany
WHO International Clinical Trials Registry Platform	Search portals to central database with links to original records. Regular fetch of trials from currently 16 data providers, including sources mentioned earlier	no	Public & free	Search Service	Worldwide
Clinical Trials Reporting Program (CRTP)	Clinical trial database	yes	Public & free	Search Service	U.S.
OpenTrials	Clinical trial database	yes	Public & free	Search Service	Worldwide

Table 13 Identified data sources for the clinical trial locator (Johannes Idelhauser 2016)

## 11.3 Appendix C

### EBM Guidelines information retrieval resources

Name	Description	API	Access	Volume
BMJ Best Practice	Evidence-based information to offer step-by-step guidance on diagnosis, prognosis, treatment and prevention.	yes	subscription	?
DynaMedPlus	Evidence-based clinical overviews and recommendations. Content updated daily. Also offers calculators, decision trees and unit and dose converters.	yes	subscription	>3,200 topics and >500 journals
EBMeDS	Platform-Independent web service CDS-System with EBM module	yes	commercial	
Essential Evidence	POC system with topics, guidelines, abstracts, and summaries of most common clinical cases. Also links to other resources like Cochrane Library and Evidence-Based Medicine Guidelines.	?	subscription	>13,000 topics, guidelines, abstracts & summaries
Medscape eMedicine /	Largest clinical knowledge base available freely. Articles updated yearly. Also available as mobile application.	no	Free, registration required	~6,800 articles
Physician Data Query	Cancer database from the U.S. National Cancer Institute. Contains peer-reviewed information on cancer treatment in the form of summaries for patients and professionals.	no	public	
UpToDate	Popular evidence-based POC tool for a wide range of disciplines but targeted on internal medicine. Extensive peer-review process to ensure accurate and precise recommendations.	yes	subscription, some articles free	~8,500 topics

Table 14 EBM Recommendation Sources (Johannes Idelhauser 2016)

## 11.4 Appendix D

### Medical publication information retrieval resources

Name	Description	API	Access	Volume
Cochrane Library	Collection of health-related databases. Its core is Cochrane Reviews, a database of systematic reviews and meta analyses.	?	Subscription	?
Google Scholar	Search engine for scientific publications of all fields. Automatically crawls many journals.	no	public & free	estimated at 160 million articles
Ovid	Science search platform that includes many databases, including MEDLINE	?	subscription	?
PubMed	Search engine mainly accessing MEDLINE database and focused on health topics. Query expansion by using MeSH ontology.	yes	public & free	>24 million records, about 500,000 new records each year
ScienceDirect	Website with access to large database of scientific publications from many fields.	yes	free (abstracts), subscription (full-text)	12 million records from 3,500 journals and 34,000 eBooks
Scopus	Database with abstracts and citations from many academic journals and many scientific fields, not focused on health topics.	yes	Paid subscription	~60 million records, >21,500 peer-reviewed journals
Springer API	Access to all Springer published journals, also includes BioMedCentral open-access publications.	yes	partly free, partly subscription	~ 2,000 journals and > 6,500 books per year, access to >10 million online documents

Table 15 Literature Service Data Sources (Johannes Idelhauser 2016)

11.5 Appendix E

Drug interaction service sequence diagram

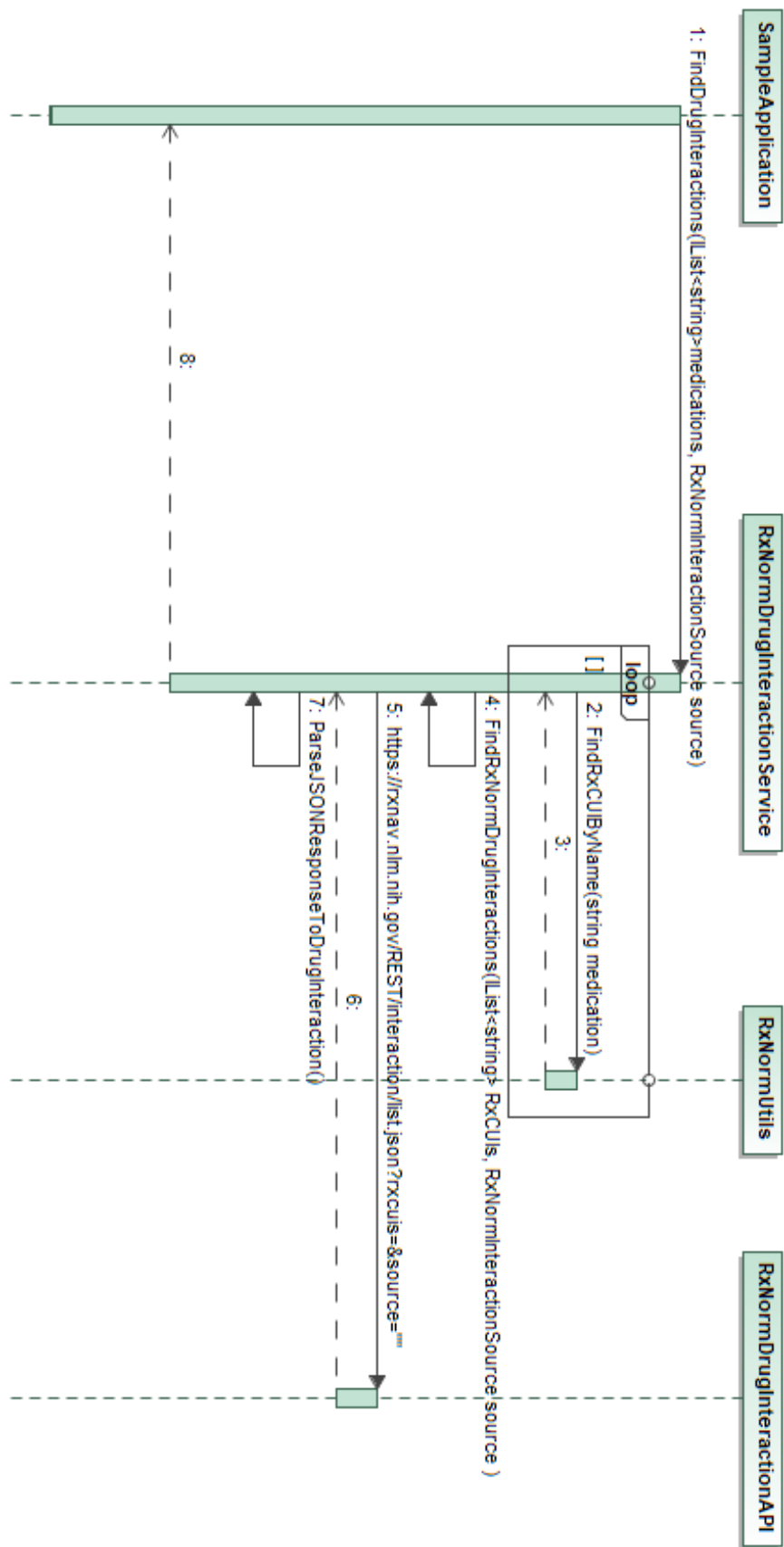


Figure 40 Drug interaction service sequence diagram

11.6 Appendix F

CDS-Services: drug adverse events service sequence diagram

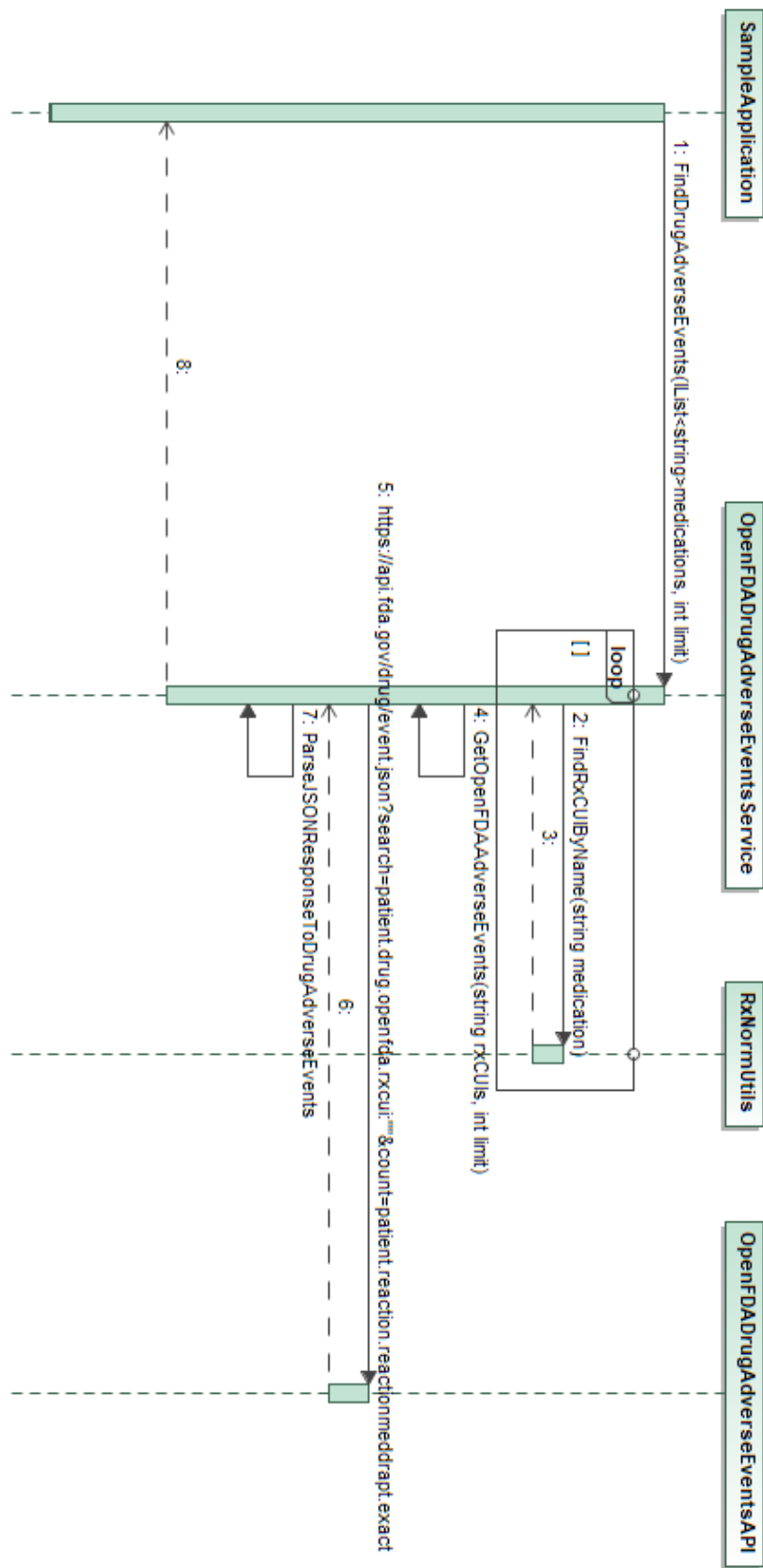


Figure 41 Drug adverse events service sequence diagram

11.7 Appendix H

SAGE-CARE Sample EHR application data model

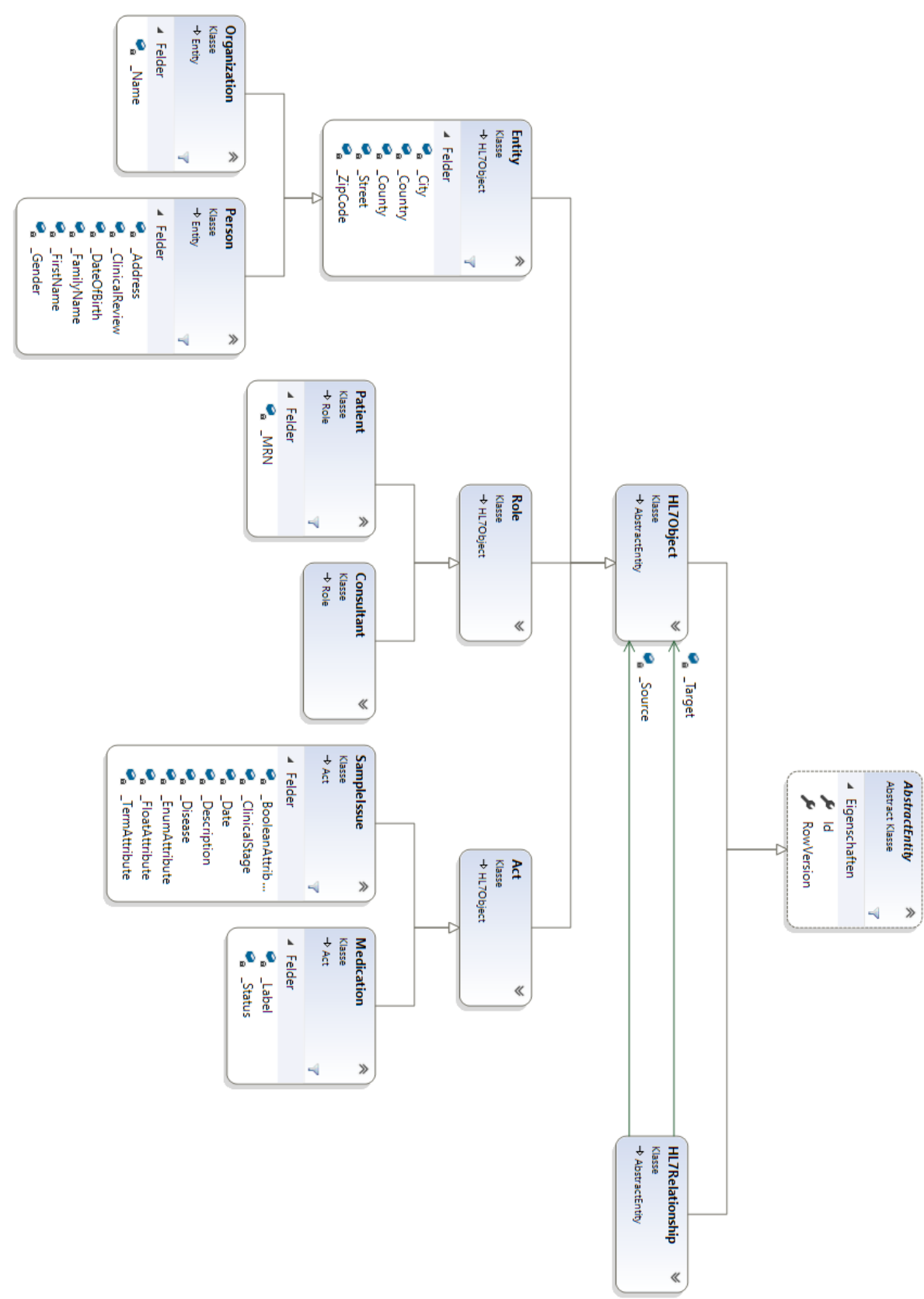


Figure 42 SAGE-CARE Information Model

## 11.8 Appendix I

### SAGE-CARE\_Client CDS-Services configuration template

```
<div class="root-composite">
  <div class="root-composite-header">
    <span class="glyphicon glyphicon-info-sign"></span>
    <strong> Clinical Decision Support Services</strong>
  </div>
  <div class="root-composite-body">
    <!-- Dashboard Panels -->
    <uib-accordion close-others="false">

      <!-- Drug Interaction Service-->
      <div class="row" ng-controller="SAGE_CARE.Components.Cds.Common.DrugInteraction.DrugInteractionController as dic">
        <div class="col-md-12">
          <div uib-accordion-group class="panel-default" is-open="dic.isopen" template-url="group-template.html" ng-click="dic.searchInteractions()">
            <uib-accordion-heading>
              <span class="glyphicon glyphicon-resize-small"></span>
              <strong> Drug Interactions</strong>
            <span class="fa pull-right" ng-class="{ 'fa-angle-up': dic.isopen, 'fa-angle-down': !dic.isopen }"></span>
          </uib-accordion-heading>
          <div ng-include="app/components/cds/common/DrugInteraction/drugInteractionService.html"></div>
        </div>
      </div>

      <!-- Drug Adverse Event Service -->
      <div class="row" ng-controller="SAGE_CARE.Components.Cds.Common.DrugAdverseEvent.DrugAdverseEventController as draec">
        <div class="col-md-12">
          <div uib-accordion-group class="panel-default" is-open="draec.isopen" template-url="group-template.html" ng-click="draec.searchAdverseEffects()">
            <!-- Box Heading -->
            <uib-accordion-heading>
              <span class="glyphicon glyphicon-alert"></span>
              <strong> Drug Related Adverse Effects</strong>
            <span class="fa pull-right" ng-class="{ 'fa-angle-up': draec.isopen, 'fa-angle-down': !draec.isopen }"></span>
          </uib-accordion-heading>
          <!-- Box Body -->
          <div ng-include="app/components/cds/common/DrugAdverseEvent/drugAdverseEventService.html"></div>
        </div>
      </div>

      <!-- Clinical Trial Service-->
      <div class="row">...</div>

      <!-- EBM Guidelines Service -->
      <div class="row">...</div>

      <!-- Literature Service -->
      <div class="row">...</div>
    </uib-accordion>
  </div>
</div>
```

Listing 22 SAGE-CARE\_Client configuration file template CDS-Systemervices.html

## HL7 RIM

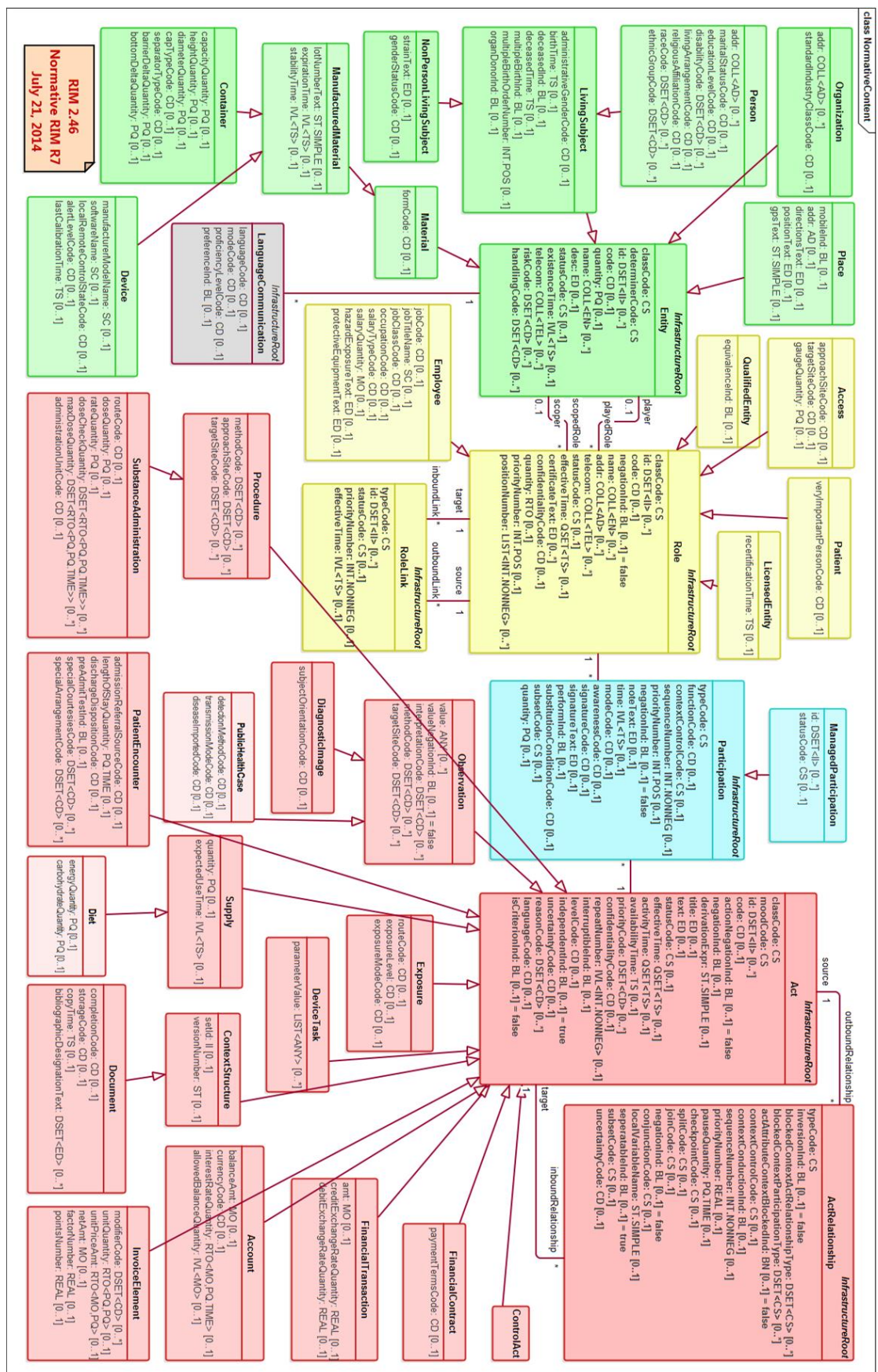


Figure 43 HL7 Reference Information Model (HL7 RIM)



## 12Publication bibliography

Advancing Open Standards for the Information Society (OASIS). Reference model for service oriented architecture 1.0 (2006). Available online at <http://docs.oasis-open.org/soa-rm/v1.0/>, checked on 7/10/2018.

Bates, D. W.; Leape, L. L.; Cullen, D. J.; Laird, N.; Petersen, L. A.; Teich, J. M. et al. (1998): Effect of computerized physician order entry and a team intervention on prevention of serious medication errors. In *JAMA* 280 (15), pp. 1311–1316.

Beez, Ulrich; Humm, Bernhard G.; Walsh, Paul (2015): Semantic AutoSuggest for Electronic Health Records. In Hamid R. Arabnia, Leonidas Deligiannidis, Quoc-Nam Tran, International Conference on Computational Science and Computational Intelligence (Eds.): 2015 International Conference on Computational Science and Computational Intelligence. CSCI 2015 : 7-9 December 2015, Las Vegas, Nevada, USA : proceedings. 2015 International Conference on Computational Science and Computational Intelligence (CSCI). Las Vegas, NV, USA, 12/7/2015 - 12/9/2015. International Conference on Computational Science and Computational Intelligence; CSCI; Symposium on Computational Science; CSCI-ISCS; Symposium on Computational Intelligence; CSCI-ISCI; Symposium on Education; CSCI-ISED; Symposium on Internet of Things & Internet of Everything; CSCI-ISOT; Symposium on Big Data & Data Science; CSCI-ISBD; Symposium on Signal & Image Processing, Computer Vision & Pattern Recognition; CSCI-ISPC; Symposium on Artificial Intelligence; CSCI-ISAI; Symposium on Parallel & Distributed Computing & Computational Science; CSCI-ISPD; Symposium on Mobile Computing, Wireless Networks & Security; CSCI-ISMC; Symposium on Health Informatics & Medical Systems; CSCI-ISHI. Piscataway, NJ: IEEE, pp. 760–765.

Berner, Eta S. (Ed.) (2016a): Clinical Decision Support Systems. Theory and Practice. 3rd ed. 2016. Cham, s.l.: Springer International Publishing (Health Informatics). Available online at <https://ebookcentral.proquest.com/lib/subhh/detail.action?docID=4613364>.

Berner, Eta S. (Ed.) (2016b): Clinical Decision Support Systems. Theory and Practice. 3rd ed. 2016. Cham, s.l.: Springer International Publishing (Health Informatics). Available online at <http://ebookcentral.proquest.com/lib/subhh/detail.action?docID=4613364>.

Bond, William F.; Schwartz, Linda M.; Weaver, Kevin R.; Levick, Donald; Giuliano, Michael; Graber, Mark L. (2012): Differential diagnosis generators: an evaluation of currently available computer programs. In *Journal of general internal medicine* 27 (2), pp. 213–219. DOI: 10.1007/s11606-011-1804-8.

BVBA, Zenjoy (2017a): Guides Project. Guides Checklist. Available online at <https://www.guidesproject.org/>, updated on 11/20/2017, checked on 8/11/2018.

BVBA, Zenjoy (2017b): Guides Project Checklist. Overview of success factors for guideline-based computerised decision support (CDS). Available online at <https://www.guidesproject.org/success-features>, updated on 11/20/2017, checked on 8/11/2018.

Clarke, Martina A.; Belden, Jeffery L.; Koopman, Richelle J.; Steege, Linsey M.; Moore, Joi L.; Canfield, Shannon M.; Kim, Min S. (2013): Information needs and information-seeking behaviour analysis of primary care physicians and nurses: a literature review. In *Health information and libraries journal* 30 (3), pp. 178–190. DOI: 10.1111/hir.12036.

Clements, Paul; Northrop, Linda (2009): Software product lines. Practices and patterns. 7. print. Boston, San Francisco, New York, Toronto, Montreal, London, Munich, Paris, Madrid, Capetown, Sydney, Tokyo, Singapore, Mexico City: Addison-Wesley (SEI series in software engineering).

ClinicalTrials.gov. Available online at <https://clinicaltrials.gov/>, checked on 7/5/2018.

Cloud-10 Multi Tenancy and Physical Security - OWASP (2018). Available online at [https://www.owasp.org/index.php/Cloud-10\\_Multi\\_Tenancy\\_and\\_Physical\\_Security](https://www.owasp.org/index.php/Cloud-10_Multi_Tenancy_and_Physical_Security), updated on 2/1/2018, checked on 7/17/2018.

Col, Nananda; Correa-de-Araujo, Rosaly (2014): Consumers and Clinical Decision Support. In : Clinical Decision Support: Elsevier, pp. 741–769.

COMM/RTD: SemAntically integrating Genomics with Electronic health records for Cancer CARE | Projects | H2020 | CORDIS | European Commission. Publication Office/CORDIS. Available online at [https://cordis.europa.eu/project/rcn/194165\\_en.html](https://cordis.europa.eu/project/rcn/194165_en.html), checked on 6/30/2018.

DailyMed. Web Services (2018). Available online at <https://dailymed.nlm.nih.gov/dailymed/app-support-web-services.cfm>, checked on 8/11/2018.

Del Fiol, Guilherme; Yu, Hong; Cimino, James J. (2014): Infobuttons and Point of Care Access to Knowledge. In : Clinical Decision Support: Elsevier, pp. 515–549.

Gacek, Critina; Anastasopoulos, Michalis (2001): Implementing product line variabilities. In *SIGSOFT Softw. Eng. Notes* 26 (3), pp. 109–117. DOI: 10.1145/379377.375269.

Greenes, Robert A. (2014a): Clinical Decision Support. The Road to Broad Adoption. 2nd ed. Burlington: Elsevier Science.

Greenes, Robert A. (2014b): Features of Computer-Based Clinical Decision Support. In : Clinical Decision Support: Elsevier, pp. 111–144.

HealthIT.gov (2018): EHR. What is an electronic health record (EHR)? Available online at <https://www.healthit.gov/faq/what-electronic-health-record-ehr>, updated on 8/8/2018, checked on 8/9/2018.

Henderson, Emily J.; Rubin, Greg P. (2013): The utility of an online diagnostic decision support system (Isabel) in general practice: a process evaluation. In *JRSM short reports* 4 (5), p. 31. DOI: 10.1177/2042533313476691.

HL7 DSS. HL7 Decision Support Service (DSS). Available online at [http://www.hl7.org/implement/standards/product\\_brief.cfm?product\\_id=12](http://www.hl7.org/implement/standards/product_brief.cfm?product_id=12), checked on 8/14/2018.

HL7 FHIR (2017). Available online at <https://www.hl7.org/fhir/>, updated on 6/7/2018, checked on 8/14/2018.

HL7 RIM. HL7 Reference Information Model. Available online at <http://www.hl7.org/implement/standards/rim.cfm>, checked on 8/9/2018.

HL7 Standards. Available online at <http://www.hl7.org/implement/standards/index.cfm?ref=nav>, checked on 8/9/2018.

HL7 vMR. HL7 Version 3 Standard: Clinical Decision Support; Virtual Medical Record (vMR) Logical Model, Release 2. Available online at [http://www.hl7.org/implement/standards/product\\_brief.cfm?product\\_id=338](http://www.hl7.org/implement/standards/product_brief.cfm?product_id=338), checked on 8/14/2018.

Hoffer, Edward P.; Feldman, Mitchell J.; Kim, Richard J.; Famiglietti, Kathleen T.; Barnett, G. Octo (2005): DXplain: Patterns of Use of a Mature Expert System. In *AMIA Annual Symposium Proceedings*, pp. 321–325.

Holst, H.; Aström, K.; Järund, A.; Palmer, J.; Heyden, A.; Kahl, F. et al. (2000): Automated interpretation of ventilation-perfusion lung scintigrams for the diagnosis of pulmonary embolism using artificial neural networks. In *European journal of nuclear medicine* 27 (4), pp. 400–406.

HSPC. Healthcare Services Platform Consortium. Available online at <https://www.hspconsortium.org/about-us/our-approach/>, checked on 8/14/2018.

HSRIC: Health Informatics (2018). Available online at <https://www.nlm.nih.gov/hsrinfo/informatics.html>, updated on 6/30/2018, checked on 6/30/2018.

HSSP. Healthcare Services Specification Program (HSSP). Available online at <http://hssp.wikispaces.com/>, checked on 8/14/2018.

Humm, Bernhard G.; Walsh, Paul (2015): Flexible yet Efficient Management of Electronic Health Records. In Hamid R. Arabnia, Leonidas Deligiannidis, Quoc-Nam Tran, International Conference on Computational Science and Computational Intelligence (Eds.): 2015 International Conference on Computational Science and Computational Intelligence. CSCI 2015 : 7-9 December 2015, Las Vegas, Nevada, USA : proceedings. 2015 International Conference on Computational Science and Computational Intelligence (CSCI). Las Vegas, NV, USA, 12/7/2015 - 12/9/2015. International Conference on Computational Science and Computational Intelligence; CSCI; Symposium on Computational Science; CSCI-ISCS; Symposium on Computational Intelligence; CSCI-ISCI; Symposium on Education; CSCI-ISED; Symposium on Internet of Things & Internet of Everything; CSCI-ISOT; Symposium on Big Data & Data Science; CSCI-ISBD; Symposium on Signal & Image Processing, Computer Vision & Pattern Recognition; CSCI-ISPC; Symposium on Artificial Intelligence; CSCI-ISAI; Symposium on Parallel & Distributed Computing & Computational Science; CSCI-ISPD; Symposium on Mobile Computing, Wireless Networks & Security; CSCI-ISMC; Symposium on Health Informatics & Medical Systems; CSCI-ISHI. Piscataway, NJ: IEEE, pp. 771–775.

ICE - CDS Framework Wiki. Available online at <https://cdfsframework.atlassian.net/wiki/spaces/ICE/overview>, checked on 8/14/2018.

Implementing a multi-tenant offering in Azure using CSP – Microsoft US Partner Community blog. Available online at <https://blogs.technet.microsoft.com/msuspartner/2018/01/26/implementing-a-multi-tenant-offering-in-azure-using-csp-part-1/>, checked on 7/17/2018.

ISO/TR 20514:2005. Health informatics - Electronic health record - Definition, scope and context. Available online at <https://www.iso.org/obp/ui/#iso:std:iso:tr:20514:ed-1:v1:en>, checked on 8/9/2018.

Johannes Idelhauser (2016): A Clinical Decision Support System for Personalised Medicine. Master thesis. University of Applied Sciences Darmstadt, Darmstadt.

Kawamoto, Kensaku; Del Fiol, Guilherme; Orton, Charles; Lobach, David F. (2010): System-agnostic clinical decision support services: benefits and challenges for scalable decision support. In *The open medical informatics journal* 4, pp. 245–254. DOI: 10.2174/1874431101004010245.

Kawamoto, Kensaku; Fry, Emory; Greenes, Robert (2014): Integration of Knowledge Resources into Applications to Enable CDS. In : Clinical Decision Support: Elsevier, pp. 819–849.

Kawamoto, Kensaku; Lobach, David F. (2005): Design, Implementation, Use, and Preliminary Evaluation of SEBASTIAN, a Standards-Based Web Service for Clinical Decision Support. In *AMIA Annual Symposium Proceedings* 2005, pp. 380–384.

Kohn, Linda T.; Corrigan, Janet M.; Donaldson, Molla S. (Eds.) (2000): To Err is Human: Building a Safer Health System. National Academies Press (US). Washington (DC).

Kuperman, Gilad J.; Bobb, Anne; Payne, Thomas H.; Avery, Anthony J.; Gandhi, Tejal K.; Burns, Gerard et al. (2007): Medication-related clinical decision support in computerized provider order entry systems: a review. In *Journal of the American Medical Informatics Association : JAMIA* 14 (1), pp. 29–40. DOI: 10.1197/jamia.M2170.

LOINC. The freely available standard for identifying health measurements, observations, and documents. Available online at <https://loinc.org/>, checked on 8/14/2018.

Loya, Salvador Rodriguez; Kawamoto, Kensaku; Chatwin, Chris; Huser, Vojtech (2014): Service oriented architecture for clinical decision support: a systematic review and future directions. In *Journal of medical systems* 38 (12), p. 140. DOI: 10.1007/s10916-014-0140-z.

Maggio, Lauren A.; Cate, Olle ten; Moorhead, Laura L.; van Stiphout, Feikje; Kramer, Bianca M. R.; ter Braak, Edith et al. (2014): Characterizing physicians' information needs at the point of care. In *Perspectives on Medical Education* 3 (5), pp. 332–342. DOI: 10.1007/s40037-014-0118-z.

Microsoft (Ed.) (2012): Developing Multi-tenant Applications for the Cloud. on Microsoft Windows Azure. With assistance of Dominic Betts, Alex Homer, Alejandro Jezierski, Masashi Narumoto, Hanz Zhang, checked on 7/16/2018.

Mike Wasson (2018): Identity Management for Multitenant Applications. Available online at <https://docs.microsoft.com/en-us/azure/architecture/multitenant-identity/>, updated on 7/13/2018, checked on 7/16/2018.

Multi-tenant SaaS patterns - Azure SQL Database (2018). Available online at <https://docs.microsoft.com/en-us/azure/sql-database/saas-tenancy-app-design-patterns>, updated on 7/13/2018, checked on 7/17/2018.

National Academies Press (US) (2001): Crossing the Quality Chasm: A New Health System for the 21st Century. Washington (DC).

NLM (2018): RxNorm. Available online at <https://www.nlm.nih.gov/research/umls/rxnorm/>, updated on 6/23/2018, checked on 8/14/2018.

NSilico: Simplicity MDT | Products - NSilico. Available online at <http://www.nsilico.com/SimplicityMDT>, checked on 7/5/2018.

OpenCDS: OpenCDS Home. Available online at <http://www.opencds.org/>, checked on 8/14/2018.

Patrick Spitzer (2016): A Dynamic Product Line for an Electronic Health Record Management System in Cancer Care. Master thesis. University of Applied Sciences Darmstadt, Darmstadt.

Phansalkar, Shobha; Desai, Amrita A.; Bell, Douglas; Yoshida, Eileen; Doole, John; Czochanski, Melissa et al. (2012): High-priority drug–drug interactions for use in electronic health records. In *Journal of the American Medical Informatics Association : JAMIA* 19 (5), pp. 735–743. DOI: 10.1136/amiajnl-2011-000612.

Pohl, Klaus; Böckle, Günter; Linden, Frank (2005): Software Product Line Engineering. Foundations, Principles, and Techniques. Berlin, Heidelberg: Springer-Verlag Berlin Heidelberg. Available online at <http://site.ebrary.com/lib/alltitles/docDetail.action?docID=10229377>.

Rodriguez-Loya, Salvador; Kawamoto, Kensaku (2016): Newer Architectures for Clinical Decision Support. In Eta S. Berner (Ed.): *Clinical Decision Support Systems. Theory and Practice*, vol. 8. 3rd ed. 2016. Cham, s.l.: Springer International Publishing (Health Informatics), pp. 87–97.

Row-Level Security (2018). Available online at <https://docs.microsoft.com/en-us/sql/relational-databases/security/row-level-security?view=sql-server-2017>, updated on 7/13/2018, checked on 7/17/2018.

SMART Health IT. Available online at <https://smarthealthit.org/>, checked on 8/14/2018.

SNOMED International. Available online at <https://www.snomed.org/snomed-ct>, checked on 8/14/2018.

SOA Features and Benefits. Available online at <http://www.opengroup.org/soa/source-book/soa/p4.htm>, checked on 7/26/2018.

Tino Landmann (2017): Evidence-Based Medical Recommendations for Personalized Medicine. Master thesis. University of Applied Sciences Darmstadt, Darmstadt.

Ulrich Beez (2015): Terminology-Based Retrieval of Medical Publications. Master thesis. University of Applied Sciences Darmstadt, Darmstadt.

Welch, Brandon M.; Loya, Salvador Rodriguez; Eilbeck, Karen; Kawamoto, Kensaku (2014a): A proposed clinical decision support architecture capable of supporting whole genome sequence information. In *Journal of personalized medicine* 4 (2), pp. 176–199. DOI: 10.3390/jpm4020176.

Welch, Brandon M.; Rodriguez-Loya, Salvador; Eilbeck, Karen; Kawamoto, Kensaku (2014b): Clinical Decision Support for Whole Genome Sequence Information Leveraging a Service-Oriented Architecture: a Prototype. In *AMIA Annual Symposium Proceedings* 2014, pp. 1188–1197.

What is SaaS? Software as a Service | Microsoft Azure. Available online at <https://azure.microsoft.com/en-us/overview/what-is-saas/>, checked on 7/17/2018.

WHO: ICTRP Search Portal. Available online at <http://apps.who.int/trialsearch/>, checked on 7/5/2018.

Wishart DS, Knox C, Guo AC, Shrivastava S, Hassanali M, Stothard P, Chang Z, Woolsey J.: DrugBank: a comprehensive resource for in silico drug discovery and exploration. *Nucleic Acids Res.* 2006 Jan 1;34(Database issue):D668-72. 16381955.

Wright, Adam; Sittig, Dean F. (2008): A four-phase model of the evolution of clinical decision support architectures. In *International journal of medical informatics* 77 (10), pp. 641–649. DOI: 10.1016/j.ijmedinf.2008.01.004.