A Large-Scale OCL Constraint Repository And Comprehensive Analysis For Supporting Automated Cancer Registry System

Imad Munir

Thesis submitted for the degree of Master in 60 credits

Faculty of mathematics and natural sciences

Institute of Informatics

UNIVERSITETET I OSLO

Autumn 2017

A Large-Scale OCL Constraint Repository And Comprehensive Analysis For Supporting Automated Cancer Registry System

Imad Munir

2017

A Large-Scale OCL Constraint Repository And Comprehensive Analysis For Automated Cancer Registry System

Imad Munir

http://www.duo.uio.no/

Printed: Reprosentralen, Universitetet i Oslo

Abstract

Cancer Registry stores cancer data collected through local cancer registries i.e., clinical department, hospitals, health communities etc. The purpose of collecting data is to understand and address the cancer disease more efficiently and effectively. According to an estimation the percentage of cancer patients in the world will rise to 30 million in the year 2020. Therefore, the data in cancer registries must be correct and updated regularly. In cancer registry numerous medical rules are defined by different medical entities. Every medical entity defines medical rules based on their requirement gathering and understanding. This makes it difficult to parse due to conflict in the ways medical entities addresses medical rules. Every country has its own National Cancer Registry; therefore, a comprehensive analysis framework is needed.

A model-based framework named Model-based Tool Analysis Framework for OCL Constraints is built for Cancer Registry System in which medical rules are defined based on the attributes retrieved from different cancer registries analysis. Validation and aggregation is done using Unified Modeling Language (UML) and Object Constraint Language (OCL). The framework captures the domain knowledge of different cancer registries and gathers attributes used in each cancer registry to create a UML class diagram and then specifies the medical rules using OCL constraints. The framework also supports the evaluation of each OCL constraint based on completeness in terms of checking whether each rule is defined using all required attributes, correctness in terms of checking each rule produces the expected output and conformance in terms of checking each rule is defined following the international standard. The results show that the framework can help in evaluating medical rules and can leads towards a comprehensive framework.

Acknowledgment

To work on this thesis required a lot of assistance and guidance. I am honored and privileged to have all along the completion of my thesis. There are many individuals whom I could not forget to thank them.

I would be grateful and indebted to Almighty Allah for providing me with this great opportunity to study in University of Oslo and to achieve my goals by completing this thesis and master degree program.

I respect and thank Shuai Wang, for providing me every insight to work in this thesis work and giving me all the support and guidance. I am extremely grateful for him for giving me appropriate time whenever I needed out of his busy corporate affairs. I would also like to pay my respect and regards to Tao Yue and Shaukat Ali for administrating this thesis and assisting me wherever I needed them.

I owe my deep gratitude to University of Oslo for giving me admission in this great institution and providing great study environment in the form of library and study rooms. I am thankful and fortunate to get guidance and experience of some of the great teachers of the institution.

At the end I may my respect and gratitude to my parents for letting me to become the person I am today and helping and encouraging me at every moment of my life. I also want to pay my heartily gratitude to my brother Fawad Munir and his wife Finza Imtiaz Minhas for providing me every sort of support for coming in Norway till the completion of my master. I have no words to express my deep gratitude to both, without their support and guidance all the way I would not have been able to complete my master degree.

Blindern, Imad Munir

1st November 2017

Table of Contents

Contents

List of Figures

List of Graphs

List of Tables

Abbreviation

Advanced E-cancer reporting and Registry Operation	AERRC
Automated Cancer Registry System	ACRS
Cancer Registry of Norway	CRN
Central Cancer Registries	CCRs
Central Population Register	CPR
Diagnostic Certainty	DS
Geographical Information Systems and Science	GIS
Identification	ID
International Agency for Research on Cancer	IARC
Model-based Engineering	MBE
Model-based Framework for Cancer Registry	MBF4CR
Model-driven Engineering	MDE
National Cancer Database	NCC
National Central Cancer Registry	NCCR
National Program of Cancer Registries	NPCR
Object Constraint Language	OCL
Object Management Group	OMG
Personal Identification Number	PNR
Rational Software Architect	RSA

The North American Association of Central Cancer registries	NAACCR
The Study of Active Monitoring in Sweden	SAMS
Unified Modelling Language	UML
United States Cancer Statistics	USCS
User Interface	UI

Chapter 1

1 Introduction

1.1 Introduction

Modeling [1] is one of the most covetable technique that is being used now-a-days. The reason lies in the advantages that modeling provides i.e., cost effectiveness, reusability, robustness, less error-prone. Modeling has been applied in many Software Engineering principles. The core reason is the extensibility of using modeling technique like UML with certain constraint languages like OCL which helps in making a system more effective and omit errors at design phase by ruling out the conditions which are not correct. The intention is to build a large rule repository containing medical rules defined by Researches of Simula when they were working with CRN [2], using design phase for error detection and correction based on constraints defined in modeling. To build such a repository much effort is required to learn, collect different information of health data along with the modeling technique of different cancer registries. How the concept of limitation is handled and what are the ways of rule selection in terms of modeling in cancer registry? First approach is collecting data by learning and understanding the domain analysis of different cancer registry and then to develop a OCL constraint repository.

Cancer [3] is one of the most challenging issue of the current era. To control and have a check on it we need to have data relevant to cancer. In this perspective, a large amount of effort is used to gather information about the treatments, diagnosis, stages, health records etc. To gain sufficient information in this area the National Cancer Registry [4] plays a key role by collecting information of all the cancer patients in the country. Such information is collected through different medical entities e.g., clinical departments, hospitals, and other cancer registries of the country. All the incoming data is checked for validity through more than 1000 medical rules.

The CRN is converted to an ICT-based ACRS by Researchers of Simula [2]. The domain analysis of CRN provides that their registry works under three principles i.e., 1) Incoming Cancer Messages 2) Incoming Cancer Messages Related to Cancer Case 3) Aggregation of Cancer Messages with their respective Cancer Cases. CRN possess challenges [5] like 1) less domain knowledge 2) completeness of medical rules 3) correctness of medical rules 4) conformance of medical rules

Cancer detection mechanism works on principles of evaluating medical rules. This means that there are a lot of medical rules in cancer registry which are considered while detecting a patient progress i.e., cancer type, diagnosis, surgery, stage etc. The research includes these medical rules written in medical terms to be converted into computing language for better optimization. My research work includes gathering information to understand the medical rules for converting them into a computing language and to build a large rule repository which contains all the medical rules. Thus conducting a complete and thorough analysis of automated cancer registry system. The advantages of using model-based approach in this perspective are as follow:

- Applying model-based approach on large scale case study implements certain benefits like the separation of core principles with design/implementation gives evolutionary development
- Product variability becomes flexible due to model reusability
- Verification of model before implementation
- Basis for other related frameworks.

1.2 Research Area

As stated earlier Researchers from Software Engineering Department of Simula proposed a model-based framework approach named MBF4CR [2]. This approach helps in capturing domain knowledge using UML [6] and applying constraints on UML using OCL [7]. This approach helps in automatic selection and execution of related medical rules. The research area lies in the analysis and evaluation of all medical rules on which constraints are applied using OCL for completeness, correctness, and conformance. Researchers from Simula defined a data set consisting of nine attributes that are used to define the medical rules in CRN i.e.,

1. Basis
2. Surgery
3. Morphology
4. Metastases

5. Gender
6. Combo
7. Toplok
8. Diverse
9. DS

The model can be used by software engineers to build cancer registry system based on their requirement specifications but the general idea would be same. In the present case using UML for capturing the domain knowledge of different Cancer Registry helps to establish a rule repository. In a software development environment business needs and development process are first considered. UML facilitate these needs and provides advance functionality and extensibility using OCL.

1.3 My Research Area

MBE [8] has become an essential approach for software development. During the recent years it has become a standard to use this approach for building models and testing them based on available limitations. This approach is also used in enterprises due to its nature of solving and capturing the domain specific knowledge of complex problems. The main three advantages of using this approach are reusability, interoperability, portability.

My research includes large-scale study of OCL constraint repository to specify medical rules and perform an analysis on these constraints.

This study will illustrate the modeling of Cancer Registry using UML [9, 10] and ensuring the correct associations, attribute relations and operations using OCL [11]. The study will also focus mainly on the evaluation of each OCL constraint based on parameters explained in chapter 4 section 4.2.4 to ensure completeness, correctness, and conformance.

1.4 Summary

The chapter concludes the importance of modeling in software engineering by defining its benefits like code reusability, robustness etc. The chapter also gives an overview of how

cancer registries work and what is the role of National Cancer Registry in collecting cancer information. It also highlights the work done by the Researches of Software Engineering Department Simula, which acts as starting point of this thesis in evaluating cancer registries. It gives an overview of the research being carried out based on the comprehensive analysis of automatic cancer registry system in terms of OCL constraints repository.

1.5 Thesis Outline

The overview of the content and structure of thesis is provided here.

Chapter 1 – Introduction

This chapter includes an introduction of modeling aspect with cancer registry and research area.

Chapter 2 – Background

This chapter includes a summary of MBE along with UML and OCL.

Chapter 3 – Problem Statement and Research Work

This chapter explains the research plan and the steps to carry out the research.

Chapter 4 – OCL Constraint Repository

The chapter briefly explains the research work done to build OCL rule repository.

Chapter 5 – Analysis of OCL Constraint

This chapter describes all the parameters used for the analysis and how it is analyzed.

Chapter 6 – Tool Support

This chapter includes the tool used to build the desktop application and how to use it by providing screenshots.

Chapter 7 – Conclusion

This chapter illustrates the conclusion of the research work and tool implementation.

Chapter 2

2 Background

2.1 MBE

A model [12] is a virtual representation of the behavior, operations and all other features related to real world. A model is used to deliver visual representation of how a system would look like and how it will behave in real world. For comprehensive analysis of automated cancer registry MBE [8] has been used which has become a major criterion in software engineering. The main advantage is the automation, high level of abstraction, continuity, generality, scalability, communication among stakeholders. Building a model is the most basic task in MBE. A model serves as communication. Therefore, MBE has been used for many system languages. It also supports specification, validation, and verification. MBE helps in improving quality and reducing development cycle of complex systems. MBE applied at upper level of abstraction defines how the upper layer model can be or must be verified and validated before going into the complex implementation of a system.

2.2 UML

UML [13] is a developmental, modeling language used for specifying software systems. It helps to standardize diagrams and provides a graphical representation of the systems' design and structure. The benefit of using UML is that it is independent of the platform. UML also supports many procedural languages such as C, visual basic etc. UML also provides the functionality of introducing new functional properties in an existing project without any complexity. It has also been standardized by OMG [14]. UML along with OCL [11, 15] enhances the system functionality at design phase by overcoming the limitations of UML.

2.3 OCL

OCL was first developed in 1995 inside IBM as an expression language [7]. OCL was integrated with UML practices by OMG in 1997. OCL is a declarative language for describing rules on UML models. OCL has become a part of UML. OCL is in used with UML which provides the consistency in checking of UML models and enables model testing under the specified rules. OCL has become a key component of MDE or MBE. OCL is used for a

variety of purposes like defining constraints on a domain specific language, constraints on UML profile (like gender description) and for UML model automation [16]. In my analysis OCL is used as a class invariant which means that a certain field can have these valid values which remains stable/same throughout the existence of the class. For example, if we have two attributes in a class A, both integer x, y and we need first attribute always be greater than 0 then its class invariant could be as follow.

$$content\ A\ inv\colon self.x\ >\ 0$$

The above statement explains for a rule in class A, the invariant (inv) defines the condition for true or false. Self means the class A from where the attribute x is taken and evaluated based on the condition specified i.e., x is always greater than 0.

2.3.1 Motivation for using OCL

Graphical representation is the first step in designing domain specific language. Graphical representation is a much better way of viewing the information including main concepts, properties, and the relationships. At this point UML [17] comes into play for designing the requirement specifications. This limits the use of expression language i.e., one cannot define the rules and constraints to verify and validate the model. Correction and fault tolerance at design level are less error-prone, issues occurring at design level are easy to manipulate and re-design. To strengthen the usage of UML, OCL plays a vital role in it [11]. OCL is used to define constraints on models but it is also used to formalize rules of new domain specific languages one designs. OCL is used for matching model-to-model or model-to-text transformation [7]. Without OCL, specifications will be incomplete and inaccurate even if one has presented it graphically using UML. It is basically used to convey the exact information of the domain to the system. OCL cannot modify the model, it is only used to express postconditions, preconditions, invariants etc. According to the gender type we can make an OCL statement based on the cancer type i.e., male can have prostate, testicular and penile cancers whereas female can have endometrial, cervical, and ovarian cancers.

2.4 Java

Java [18] is a programming language released by Sun Microsystems in 1995. It is used for many different development purposes one of which is building desktop applications for a specified number of individuals. In this thesis a desktop application using JavaFX which is an XML-based language designed to build user interface is created. JavaFX is a java alternate for using swing. Although it is not much advanced then swing but it is gradually increasing its UI libraries with advance features and supports. The research consists of evaluating and analyzing different OCL constraints for checking software engineering principles i.e., quality of medical rules in terms of completeness, correctness, and conformance.

2.5 IBM RSA

IBM RSA [19] is a modeling, design, and development tool. This tool provides the interaction of domain specific language with UML. The primary advantage of using RSA is that it uses UML for designing java applications providing all the important features available because it is based on Eclipse Modeling Framework [20]. It was first started in early 1990s and combined with modeling approaches which was later called as UML. It also provides functionality of applying OCL constraint on UML class diagram.

I used this tool for converting the medical rules into OCL constraint where verification and validation is done on created UML class diagram. RSA tool helps in the transformation of UML to Java specification as shown in figure 5. The tool only allows transformation if the created model is correct by stating the errors. This transformation helps in checking the OCL constraints by writing code in java and executing them on UML model.

Every OCL statement in RSA starts with a “self” keyword. The keywords used in RSA tool are

- Self
- implies
- end-if
- post

- pre
- in
- inv
- let

There are many other keywords defined but these are the ones which were used frequently. Creating OCL constraint using these keywords and running the model by applying constraints verifies the medical rules.

The main nine attributes as described earlier in chapter 1 based on which medical rules are defined and inter-related are converted to OCL using the above keywords.

2.6 Summary

This chapter gives a background knowledge of all the concepts and tools that are being used. This chapter explains how MBE has been used in solving complex enterprise solutions and how much this technique has been helpful. This chapter also explain modeling language like UML and what are its benefits of using with OCL for overcoming the limitations. Java is the programming language and IBM RSA is the tool that is used along with another tool named IntelliJ that is explained in chapter 6. A brief introduction of these languages and tool and how they will be used in the research area.

Chapter 3

3 Problem Statement and Research Work

3.1 Problem Description

The task is to build a framework for OCL constraint repository not only for a specific cancer registry but could act as a starting point for any cancer registry. In simple words, a model that can be applied to any cancer registry but with different rule repositories, requirement specifications, and any other constraints that are available in it.

Now every Cancer Registry has its own way for registering incoming cancer information. Some registries store cancer information as cancer messages summarizing them based on cancer cases, some stores cancer information as complete message, therefore, each cancer registry has its own cancer attributes, constraints for verification and validation of cancer data. Our scenario is to compare the different cancer registries, build a general model containing attributes defined in different registries and creating a OCL constraint repository of medical rules.

Thesis includes the analysis and evaluation of each OCL constraint as describe in chapter 1. The research area is to check every OCL constraint for achieving the property i.e., completeness, conformance, and correctness irrespective of redundancy and long OCL statements. Concerned area is the inclusion of every possible clause in OCL statement for medical rules in detecting the type, state, surgery, diagnosis of a cancer patient for better treatment and research. Research work include,

- Understanding the domain of Cancer Registry of Norway and the existing cancer registries
- Build a large-scale repository of OCL constraints
- Conduct a comprehensive analysis based on the created case study (Chapter 4)
- Implement tool support (Chapter 6)

3.1.1 Understanding the domain of Cancer Registry of Norway and the existing cancer registries

Domain Analysis of CRN

There are two main concepts in CRN [2] based on their domain model 1) Cancer Messages 2) Cancer case. Cancer message includes the variables required for registering. The cancer information is collected through a number of sources like health sectors, hospitals, clinical laboratory, National Community of Health [5]. A cancer case divides the incoming cancer messages according to the type of cancer. Both these entities share a common set of fields which plays an important role in defining rule selection.

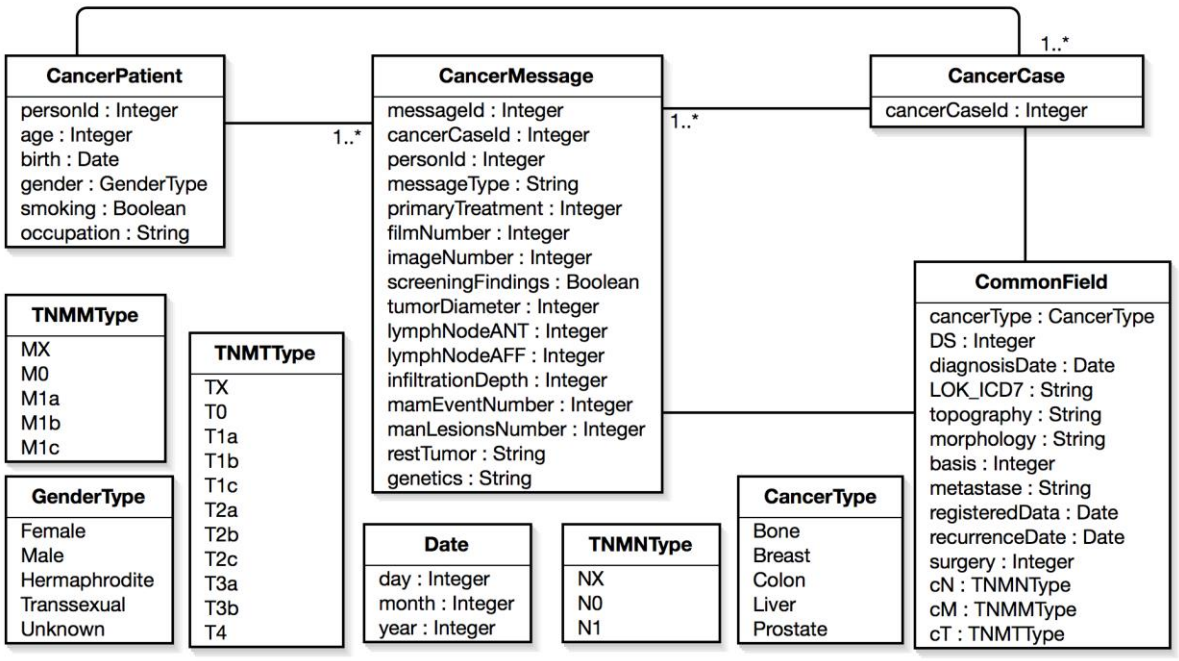


Figure 1 UML class diagram of CRN from [2]

For example, *DS* [2] exist in both cancer messages and cancer cases, which is used to determine a cancer if the value of *DS* is more than 3 (lower values of *DS* indicate a pre-cancer).

ID identifies all these different classes respective to each case. Along with ID, common fields also help to define certain relationships and properties and acts as a base for defining medical rules and using OCL tagging mechanism.

Simula used OCL [2] to define the medical rules along with the tags. This strategy later helps a lot if a change, update, delete or any database operation takes place. Previously if a rule is changed then the developer must change it from every occurrence of that rule. This approach requires effort and cost which in turn is a greater challenge. Using UML with OCL [11] Simula build an Automated Cancer Registry System where every rule is written once and changes takes place at one place only using OCL tags.

OCL [15] rules based on the domain knowledge are divided into three categories 1) Checking incoming Cancer Messages 2) Checking Cancer Case Rules 3) Checking Aggregation Rules. First level checks the correctness of incoming cancer messages. Second level checks the validation rules for cancer case to include cancer messages according to their type. Third level checks again for better performance whether cancer messages and cancer cases are aggregated without errors, warnings, or threats.

NAACCR

According to the American association the main goals for registering cancer cases are [21] 1) the data standard must be defined, 2) must be at systematic level i.e., providing education and training, 3) Population-based cancer registration, 4) Aggregation of data-sets of different health sources, 5) Promote and encourage people to work in this area.

NAACCR comprises of a dataset [22] which includes a list of variables needed for the minimum requirement for collecting cancer data along with certain elements essential for certain cancer cases. The standard of NAACCR [23] is followed by many cancer registries but the dataset is different. This means data for incoming cancer messages could be same but their used attributes could be different. Most hospital-based registries overlap with datasets form different cancer registries to retrieve data. NAACCR uses code categories to distinguish simple data and the data which overlaps with different cancer registries. These approach intents to make similar data in every registry to work in a same way and to avoid inter-registry comparability. To improve data consistency, code categories are used along with technical rules that describe when to start a treatment? How to test and add a patient? Use of data standards refers to improve the quality, consistency, and comparability of difference communities. NAACCR used Data Edit technique [24] [25] to restrain patient entry while testing with rules that are logically false. For example, it stops from checking whether a man

is suffering from cancer related to female reproductive system. Logically a man cannot have cancer of a type that is not entirely related to its gender.

NAACCR [21] has diverse group for cancer registries. Every registry department has different task for example collecting patient data or basic descriptive data. According to NAACCR data standards defined mostly depends upon some strictness level i.e., must, should, may.

NAACCR uses IBM Unified Data Model for scoping the model [26]. This approach provides the reusability of the model by mapping it with Business Data Model. IBM includes variables [26] i.e., 1) Attributes 2) Entities 3) Diagrams 4) Packages.

Cancer registry Data Element
Patient Name (first, middle, last)
Patient Social Security Number
Patient Date of Birth
Patient Sex
Patient Address Current
Patient Race
Reporting Facility ID
Primary Site C
Date of Initial Diagnosis
Class of Case – Groups cancer cases into Ana lytic vs Non-Analytic
CS Mets at DX- Identifies if there are metastases involvement at time of diagnosis
RX Date Chemo – Date Chemo Started
RX Sum – Chemo – Code for type of chemo administered
Rx Chemo Flag – Code indicating why no date has been identified

Table 1 Modeling Dataset in NAACCR using IBM Unified Data Model [26]

This is a representation of the attributes used for modeling the incoming dataset in NAACCR.

There is only a case of incoming cancer messages irrespective of their types.

Relationships to be added
Party > Patient > Patient Dimension
Party > Organization > Provider Dimension
Party > Practitioner > Practitioner Dimension
Common > Calendar Dimension
Diagnosis > Diagnosis Code Dimension
Patient Medication > Medication Dimension
Patient Medication > Medication Class Dimension
Patient Medication > Formulary Dimension

Table 2 Relationships of NAACCR [27]

NPCR

This is a program [28, 29] which collects information regarding cancer related data from different states. The mode of collecting cancer registry information is population-based. The purpose of this technique is to achieve data quality and consistency as it will only be conducted on a certain population. This program gets updated data every year by validating and verification of data standards as approve by the state.

Congress established this program [30]. It includes certain variables/attributes for cancer information like, Type, Extent, Location, Initial treatment, Result of treatment, Success rate etc. This program is currently working in 46 different states gathering more than 96% of valid data [28]. Officials that are included are Health professionals, Researchers, Medical community, Policy makers. All the information transferring from hospital or other health sectors to cancer registries is stored using standardized codes.

This program has used several freeware software for data collecting like [31]

1. Registry Plus

2. NPCR-edits

This tool comprises of three main tasks

1. Checking data quality
2. Checking data standards when collecting and aggregation
3. Preparing data for analysis

NPCR mostly use AERRC [32] tool for capturing, storing, analyzing data. This uses SAMS to identify and derive functions with the help of professionals and stakeholders with experience. Modern business model techniques are used. UML is used as modeling technique which describes stakeholders and all the connected actors, business use cases, models, and flow diagrams, updating models per the customization.

Cancer Registry of Pakistan

Earlier in Pakistan there was no system for collecting information [33]. First ever report generated was by Karachi Cancer Registry. It is population-based cancer registry providing 9 years proven data i.e., from 1995 to 2003 [34]. This report provided very important information about the percentages of cancer occurring in male and female. According to the report, males mostly suffered from lung cancer due to smoking and other tobacco substitutes and in females' breast cancer is the most common.

In Pakistan for defining standards in elicitation of cancer information IARC [35] format is used. Shaukat khanum cancer hospital Lahore is the one that has been working for the last 19 years. It is hospital-based cancer registry [36]. In compliance with IARC Shaukat Khanum also uses National Punjab Cancer Registry Form [37].

Limitations for population-based as no latest record of population present. Another limitation is the status unavailability of a new patient.

Cancer Registry of USA

In USA Seer program [38, 39] is widely used for gathering cancer information. The process takes place as data from different hospitals and clinics is collected and analyzed and send to cancer registries where data is validated and verified using national standards and state policies. The data is then sent to seer program which originates population-based statistics based on new and old data. The data showed can be divided into several ways for example statistics can be shown per the race, ethnicity, sex, age, cancer type etc.

All the information is provided and arranged by National Institute of Cancer America [38]. This institute uses Seer program along with GIS [40] to locate exactly the amount of cancer patients in a certain area.

Cancer Registry of China

Cancer data in China [41] is collected through several sources like local hospitals, clinics, urban medical institutes, rural medical implementation schemes. NCCR [42] evaluates, collects, and analyze data per the rules provided by Chinese Cancer Registration and implemented using IARC format [42]. Data from different registries is available based on population through NCCR.

As population is increasing day by day a full complete population-based analysis of cancer data is still not possible. Data [43] is available for small area of regions. Only 6 to 7 percent of total population is analyzed by NCCR. Local registries contribute to even 45 percent in some regions, but the data is still not complete and fully available.

Cancer Registry of India

In India elicitation of cancer data [44] is achieved using two approaches [45]

1. Population-based
2. Hospital-based

Cancer registration in India is followed by NCRP. According to a journal [44] NCRP has 26 population-based registries and 7 hospital-based registries data. This data includes only a

small portion of Indian population. The major concerns of Indian Cancer Registry are the authenticity of cancer diagnosis data and the updating of cancer registries.

In India, roughly there are 2.5 million cancer cases each year out of which 0.7 million are new cancer cases. Initially the population based registries started from Bombay in 1964.

Cancer Registry of Denmark

Previously data collection and registration were tumor-based [46-48] i.e., 1) date of birth and 2) name of patient. This technique is taken by PNR. The Danish Cancer Registry dataset includes clinical hospital departments, pathology departments and death certificates. During gathering of data from different medical and non-medical sources they verified the patient through PNR by running it over CPR. This approach provides benefits of 1) person is a residence 2) helps in integrating medical history coming from different sources.

They use PNR as a key attribute while modeling the system of cancer registry. If the patient PNR is not already registered, then it is a new case otherwise update the new information to the existing case. There is no definite division of cancer messages and cancer cases.

Limitation or the core issue is the registering of information in different cancer registries. Using PNR to aggregate and combine from several sources emerges the problems of data quality and correctness.

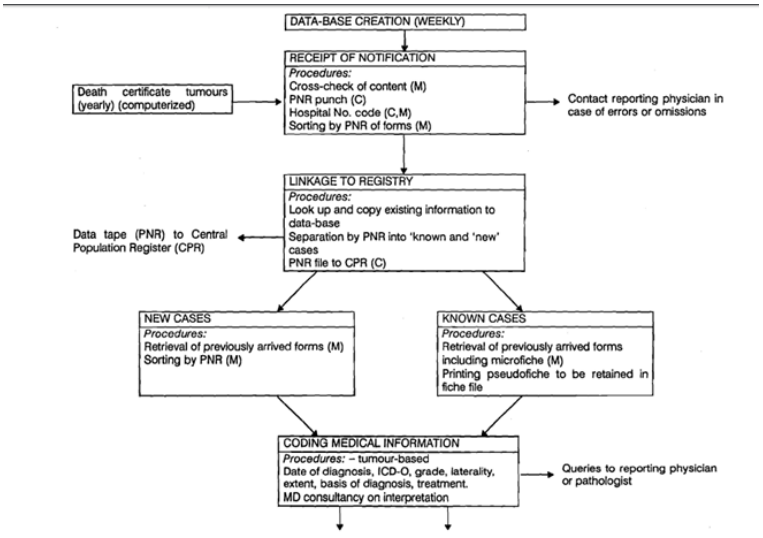


Figure 2 Working model of Denmark Cancer Registry [46]

Country	Cancer Registry	Approach	IARC Standard *
Norway [2, 5]	Cancer Registry of Norway	Population-based	Yes
North America [21, 29]	NAACCR, NPCR	Population-based	Yes
Pakistan [33, 34, 36]	Karachi Cancer Registry, National Punjab Cancer Registry	Population-based, Hospital-based	Yes
USA [38, 40, 49]	National Institute of Cancer, USCS	Population-based	Yes
China [41-43, 50]	National Central Cancer Registry	Population-based	Yes
India [44, 45]	National Cancer Registry Program	Population-based, Hospital-based	Yes
Denmark [46, 48]	Danish Cancer Registry	Population-based	Yes

Table 3 Summarization of Cancer Registries

The table summarizes the cancer registry that I have used for the comprehensive analysis. My supervisors worked on CRN, therefore I had understanding about the working model of CRN. I included Scandinavian, American, European, and Asian cancer registries. Comparing CRN and Danish Cancer Registry provides a comprehensive overview of how the cancer registry is working in the Scandinavian using which attributes. The NAACCR and NPCR cancer registry used by North America are also mostly used by many of the European cancer registries. The cancer registries [34, 36, 41, 42, 44] in Asia including Pakistan, India, and China, each of them has their own central cancer registries for data collection. The scope of data collection is not as big as like Europe and America, but it is improving day by day by collecting patient data following populations and hospital-based approach. All the registries implement international standards of data manipulation i.e., IARC. In all the cancer registries completeness and correctness are the core challenges in terms of data quality, therefore, a framework in required to address medical rules repository.

*** IARC**

IARC is an international agency for cancer research in cancer prevention [51]. The main goal of this agency is to collaborate international researches from different countries of the world to prevent cancer. This agency provides a platform where international researchers, medical

entities can interact with each other and help in diagnosing the early symptoms of cancer based on the information they have calculated in their country. IARC also works with WHO to collaborate for example like providing necessary precautions and education to the people, vaccination for early treatment of cancer symptoms.

3.1.2 Build a large-scale OCL Constraint Repository

Comprehensive domain analysis of different cancer registries in section 3.2.1 helps in understanding the attributes defined in each registry, based on which a UML class diagram can be created easily. UML helps to visualize whether every requirement specification is filled in or not. The purpose of building a comprehensive framework is to have a defined way of building cancer registry therefore, having a large-scale OCL constraint repository including cancer coding rules specified with OCL. OCL overcomes the limitations of UML i.e., enhances the functionality of model by applying constraints on them. The purpose of using OCL is that it does not changes the analogy of working model but enhances the working approach of the model i.e., in our case every cancer registry can use OCL according to their rules and regulations, but the model remains the same. Domain knowledge of different cancer registries will help us in building a case study which will explain the difference in the approach used by cancer registries. We will use the above case studies in section 3.2.1 and build a large-scale OCL constraint repository. Large-scale case study will include different dimensions i.e.,

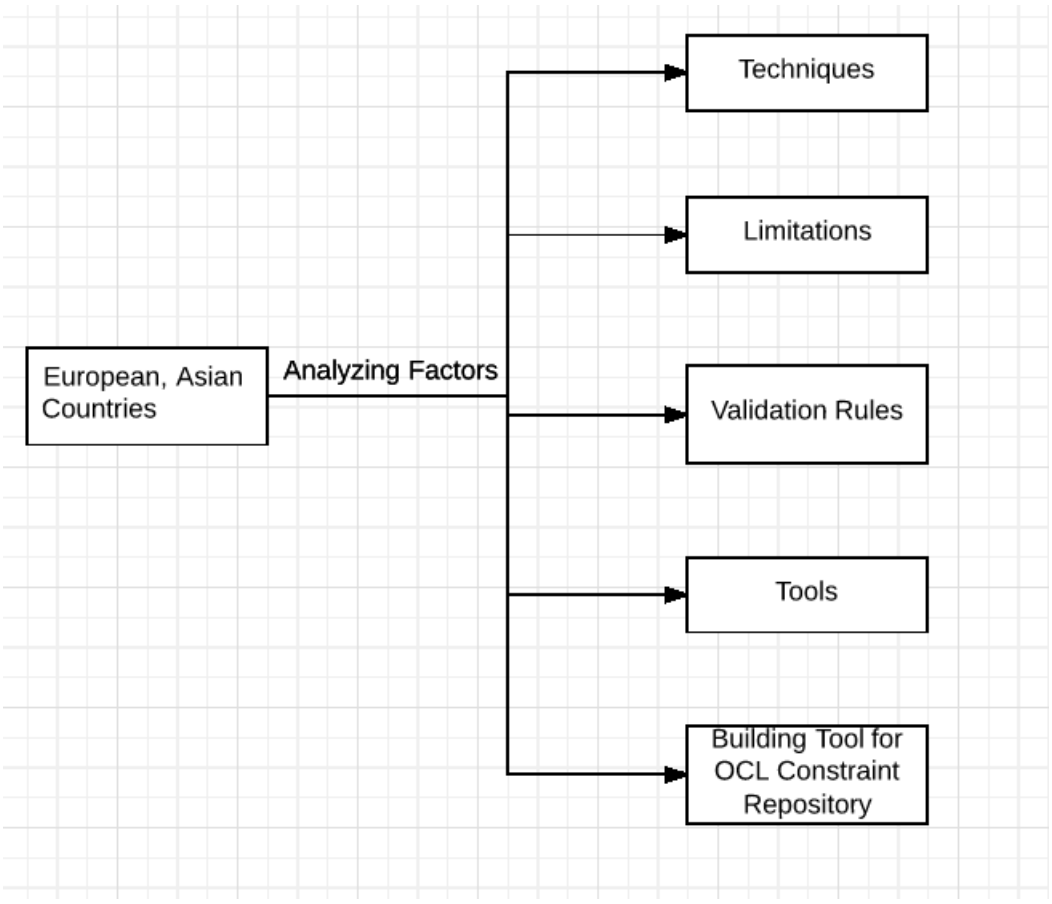


Figure 3 Research flow based on factors

1. Information elicitation though different European, Asian countries.
2. Technique these countries are using.
3. Limitations they have in collecting, merging data.
4. Validation rules that are being used.
5. Tools which they are using
6. Building a tool for OCL constraint repository

The main upcoming challenges for collecting information are

- To collect detailed information of cancer registries using different medical rules

- Authenticity of gathered medical rules and data. This means we need to have a medical expertise present whenever we model a new rule into the rule repository.
- To build a model that can be applied on any cancer registry
- Data unavailability i.e., attributes not clearly defined

Cancer data is very confidential and private therefore the availability of data and the attributes that are being used cannot be accessed in every cancer registry explained in section 2 above. Using the information that is available and thorough understanding of domain knowledge of different cancer registry a general model is created based on which a comprehensive analysis is done.

Once we have built a large-scale OCL constraint repository we will analyze it by applying different parameters for evaluation of each constraint. For parameters, we will check it under different directions like [52] 1) Completeness of OCL statements. 2) Correctness of OCL statements. 3) Conformance of OCL statements. In cancer registry the main concern is the medical rule to include all necessary variables and values for cancer data. As stated earlier NAACCR is widely used cancer registry, one of the main task of this cancer registry is the completeness of information they collect and analyze [51]. This mean that each medical rule irrespective of how long it is, how many variables it includes it must be complete and according to the standards to be fully correct. In this thesis the core analysis is to check whether each rule is complete including all the attributes based on the rules formalized by Researchers at Simula.

3.2 Summary

This chapter explains why and how the research is done. The problem statement summarizes the fact that every cancer registry has its own approach, technique of collecting cancer patient data and indeed there is a need to have a comprehensive framework for analyzing every cancer registry and create medical rules using UML and OCL. The second part of the chapter shows in detail the research done on various cancer registries and summarizing the information gathered as shown in table 3. The research shows how the general attributes are used in each registry and population and hospital-based are mostly the two approaches that are

used. Every cancer registry of a country has its National Cancer Database Center which collects data from hospitals, clinics, laboratory etc.

Chapter 4

4 OCL Constraint Repository

General purpose of cancer registry system [53] are

- Maintenance of cancer incidence reporting system
- Further research
- Helping public health sectors and agencies

There are three types of cancer registries [4]

1. Facility-based registries
2. Specialty registries
3. Central cancer registries (CCRs)

Facility-based registries normally collects information about a patient treatment or diagnosis of a facility in which he/she is. Specialty registries collects information about a certain type of cancer for example, a registry which collects information regarding only lung cancer. Third type of registry is a general registry but based on geography, central cancer registries collects information of a particular geographical area following population-based approach [4].

In cancer registries patient information is normally collected using two approaches i.e., hospital-based, and population-based [36, 37, 54].

The research statement includes a large-scale OCL constraint repository which is explained in Chapter 2 and Chapter 3 respectively, and comprehensive analysis for supporting automated cancer registry system. The research is divided into four different phases stated as in chapter 3 section 3.2.

In Europe and America NPCR and NAACCR rules are used in all the cancer registries [28-30, 54, 55]. The above pair follows almost the same attributes for information gathering [21] i.e., 1) Demographic-based 2) Historical-based 3) Diagnosis Stage 4) Cancer Stage 5) Treatment 5) Follow up.

To collect information, the model must be able to get data from incoming cancer messages as well as from other cancer registries for better treatment. Information must be [56] 1) pertinent 2) accurate 3) complete. Secondly, the information system will be designed for 1) collection 2) storage 3) management 4) analysis. NPCR and NAACCR uses National Cancer Database [3] for better information collection.

The three main concerns of the cancer registries are completeness, conformance, and correctness of all the medical rules defined in a cancer registry [57, 58]. In medical terminology, to detect the right cause of cancer one need to evaluate certain tests that gives different values and based on those values the type of cancer is defined. The research part includes firstly to convert all the medical rules written in python provided by Researchers from Simula and an excel file containing the formal medical rules into software engineering practices using OCL. Secondly, to evaluate every medical rule converted to OCL in terms of completeness, conformance, and correctness irrespective of any other functional or non-functional properties. The concern is no matter how long is the medical rule defined in OCL, it should hold property of completeness, conformance, and correctness irrespective of throughput or latency value.

4.1 Completeness, Correctness, and Conformance

In Software Engineering [59] completeness means the provided context is accordance with defined specifications i.e., does a system satisfies all the requirements and whether those requirements are feasible or not. In my thesis, the concept of completeness is used to check the availability of all the clauses required to make an OCL statement from a medical rule. This satisfies the property of completeness by applying certain formulas that are being used in this thesis paper.

Correctness means OCL constraints are correct with respect to the specifications defined. In computer science, functional correctness is based on actual output vs expected output and it is asserted that the expected output will be true. In the present scenario measuring correctness of each OCL statement using formulas defined in this thesis paper states how much a statement is correct according to the specifications and what is the difference in the value obtained.

Conformance means to check each OCL constraint whether it complies with the specification defined or not i.e., medical rules. In this context conformance of OCL statements depends

whether the OCL statements hold the property of correctness and completeness. The formulas that are used for calculating conformance also contains the parameters of completeness and correctness therefore, conformance value relies totally on the values obtained from completeness and correctness of OCL constraints.

4.2 Research Work

Cancer is one of the biggest dilemma increasing in percentage every year. Medical Institutes, researches, doctors everyone is trying to find the best suitable cure. The core supplement of their efforts is the research that has been carried out or is happening. To have the domain knowledge of cancer registry, studied different registries as explained in chapter 2. As stated earlier a model-based framework is used i.e., UML for designing a cancer registry and OCL for applying constraints on those UML class diagrams. Using OCL and Java side by side enhances the implementation at design level and development level.

The complete work flow is as follows:

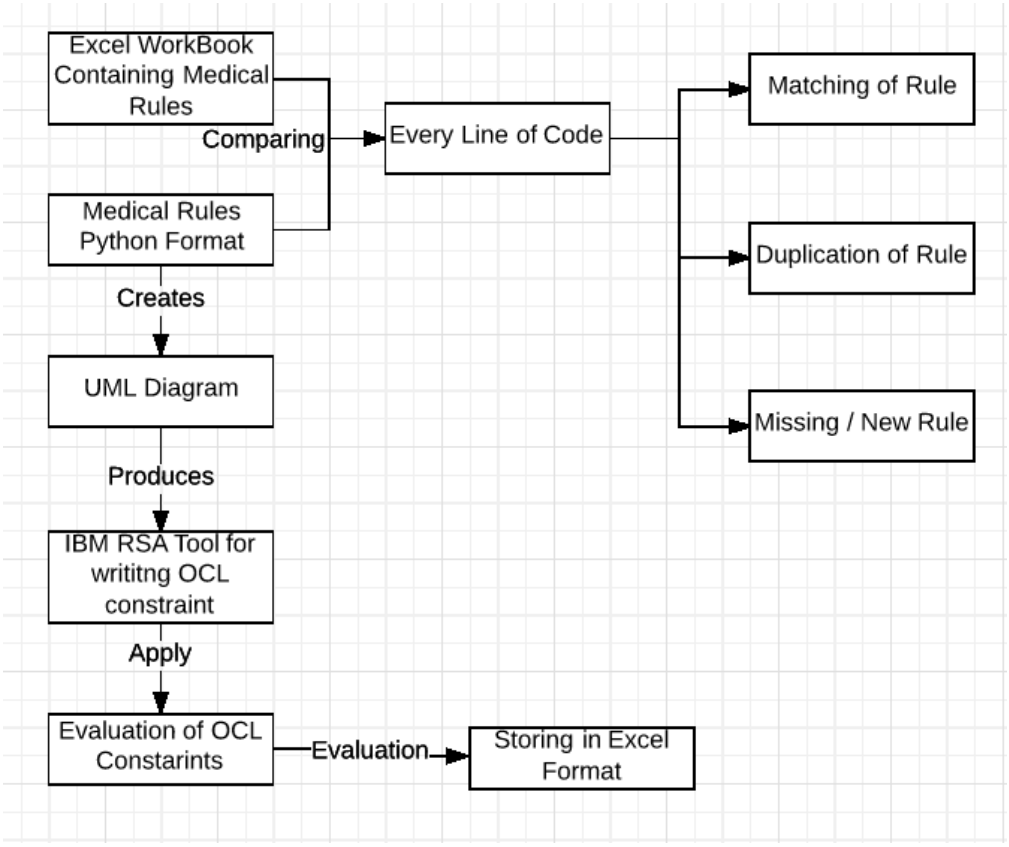


Figure 4 Work flow comparing medical rules

4.2.1 Excel Workbook

Excel file containing medical rules serves as a starting point for this research. The data set provided by Researchers from Simula containing excel and python files consisting of all medical rules. On the CRN website [60], they have provided a general pdf document named document variable in which all the cancer attributes along with their properties, relationships, tables of information are available. The attributes defined in this document are used to define the model of a cancer registry. Comparing excel file of medical rules with this document to understand and verify attribute values that are being used. The excel workbook contains several sheets named in accordance with nine attributes shown in chapter 2 before which includes the prerequisite values along with other values that should be present to make it a valid medical rule. For example, there is rule mentioned like if a patient surgery value is equal to 35 then the basis value for that patient should be from these [57,70,72,74,75,76,79,98] array values. These are just numbers that helps doctor to recognize the previous and the present state of a patient (Long history of patient). Research work include first to understand what these variables are? and how they are used? and how to make OCL statements later? Preliminary task was to study and compare the excel sheet rules with document available on the website to understand what each attribute means and how they are evaluated.

4.2.2 Python Files

Got some rules from the supervisors who were working on the CRN, written in the python format. The Ultimate task was to understand the logic of every medical rule in python file and search for any duplication, matching or missing / new rule as shown in figure 4 above. Going through every single line of code and comparing it with excel file of medical rules, this technique was based on two factors:

1. Compare python file with medical rules and search for duplication of medical rule in python file.

2. Compare python file with medical rules for any missing or new medical rule.

To achieve above two goals following loop was used for each python file shown in figure 4:

- Going through every line of code.

- If there is a match of code with medical rule write the name of medical rule on that block code and highlight it in green color.
- If there is duplication of code mentioned the medical rule and highlight it in red color.
- If there is missing or new code mentioned, it.

Going through python files helps in understanding the logic and meaning of each medical rule. The next step is to convert those medical rules using excel and python files into OCL using software engineering principles [1, 8, 11]. The tool used for this conversion is the IBM RSA version 9.1.1 [61].

4.2.3 IBM RSA

IBM RSA [61] is an old but very productive tool for domain specific language. Main advantage of this tool is embedded UML modeling tool including all features, relations, properties etc. Using UML modeling tool for building the model of cancer registry with relations, attributes, properties, and operations. RSA allows to evaluate UML model using OCL by applying constraints and verifying it based on actual vs expected value. This approach can be achieved automatically and manually. In this this project, manual approach was used in which all the medical rules were written in OCL expression. There are mainly five classes that created in RSA as shown in figure 6. After getting the domain knowledge of CRN and other cancer registries around the globe, comparing the excel and python files generated a class diagram containing 52 different attributes with associations between them. RSA helps in generating UML-to-Java transformation of created class diagram. If this step fails, UML model is not build correctly and vice versa otherwise. For example, after UML-to-Java transformation the medical rules mentioned above a patient surgery value is equal to 35 then the basis value for that patient should be from these 57,70,72,74,75,76,79,98 array values can be evaluated using java coding.

After modeling of cancer registry, next step is to create OCL statement for each medical rule. RSA provides the functionality of applying OCL constraints on UML class diagram attributes. To create a complete and precise model using model-based engineering we need both UML diagrams and OCL expressions. UML diagrams are essentials for representation of classes and associations whereas OCL works only on nonexistent elements. In RSA, one can easily

apply OCL expression only by clicking on the respective UML diagram and verify it by running OCL constraints on it. For example, from the dataset of rules obtained from supervisors working on CRN, one of the main validation step is the cancer messages validation, there is a medical rule in CRN stating that for surgery value a patient *messageType* (attribute of UML diagram) must be K or R or H. Applying this rule using OCL would be very simple, the OCL statement is:

self.surgery=07 or self.surgery=09 or self.surgery=35 implies self.cancerMessage.messageType='H' or self.cancerMessage.messageType='K' or self.cancerMessage.messageType='R'

In OCL [7] self means the class that is being used along with the attribute name and value. Implies means if the expression before it holds true then the expression after it must be true. RSA tool provides a built-in feature of UML to Java transformation. In this phase, the created UML class diagram is automatically converted into java. The benefit of using this feature is the validation, verification, and fault tolerance of UML diagram. If the transformation is successful, then model is correct otherwise incorrectly modelled. Following is the transformation of UML to java specification.

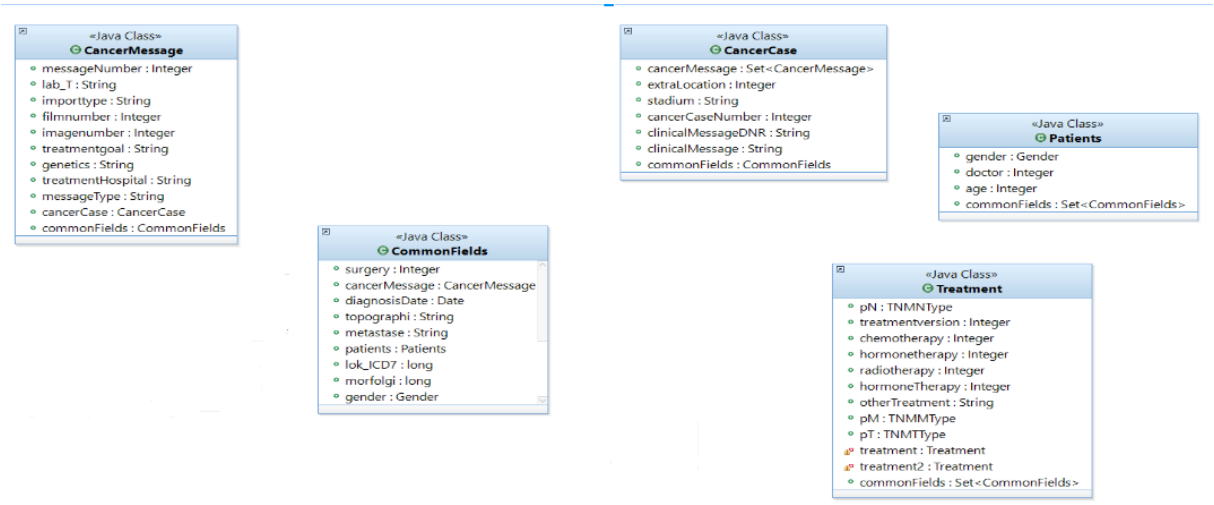


Figure 5 Java Specification of UML class diagram

Figure 2 shows the transformation of UML class diagram to Java specification. The figure shows how different classes are divided and what variables it contains. Generation of this Java specification states the fact that the created UML model is correct and therefore, it can be used for verification and validation of different medical rules. The scenario is if the created medical

rule in OCL language and the java conversion of that rule using OCL library works as expected then the UML model is correct otherwise there could be any type of error while modeling like associations wrongly define or OCL statement is not written correctly.

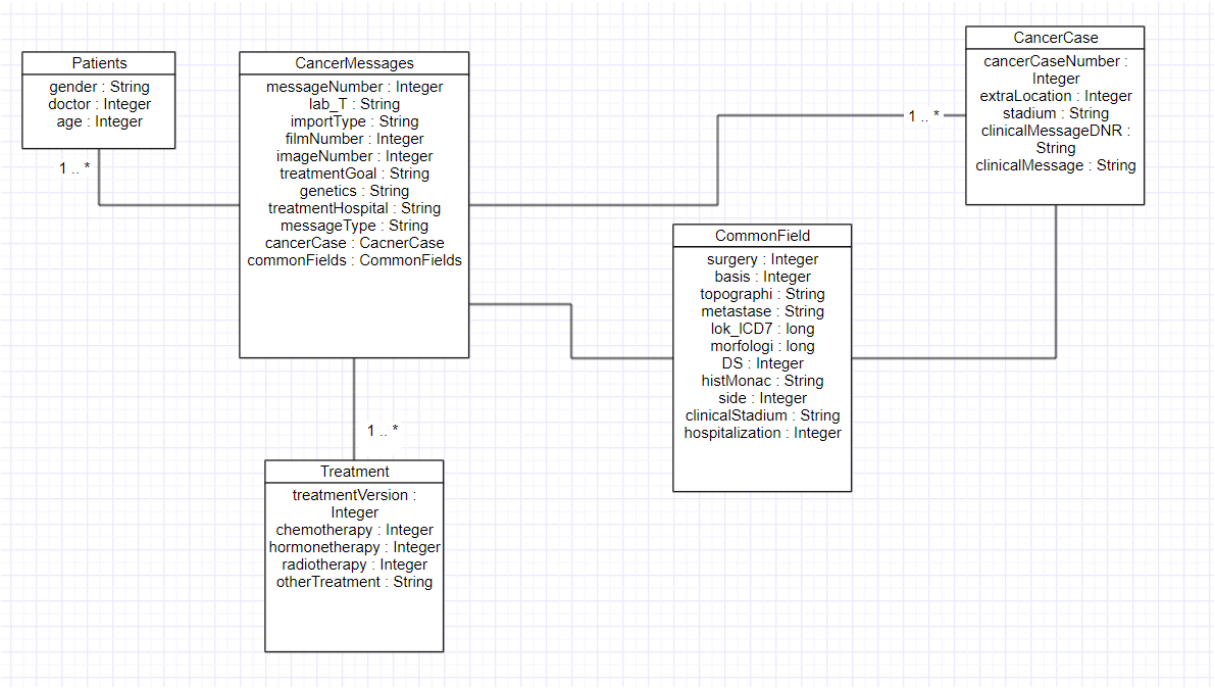


Figure 6 UML class Diagram of Cancer Registry

Figure 3 shows the UML model of cancer registry created using attributes mostly from the dataset provided by Researchers from Simula, remodeled according to the research plan. The figure describes all the attributes, classes, and the associations between them. *CommonFields* is the main class containing the attributes required to interact with all other classes. The relation between *CancerMessages* with *Patients*, *Treatment*, *CancerCase* and *CommonFields* classes is one to many. *CancerCase* has one to many association with all the classes. *CommonFields* class contains the attribute based on which most of the medical rules are defined.

4.3 Summary

This chapters explains how the research work is done. The first section of this chapter gives an overview of purpose and the three types of cancer registries. This section also explains how the two cancer registries NPCR and NAACCR corelate with one another. In the second section, completeness, correctness, and conformance are defined according to software engineering perspective and how they are used in this research work. The last section provides a framework based on which the evaluation of all OCL constraints is achieved. IBM RSA is the tool used for conversion of all medical rules into OCL expression. The benefit of using this tool is the less effort required due to UML-based modeling support and reduces complexity and therefore, increases quality and efficiency.

Chapter 5

5 Analysis of OCL Constraint

The main area of research is the evaluation of OCL statements based on the parameters defined below in table. Simula Software Engineering Department build a tool based on MBE [2], as stated earlier, following this technique the verification and validation of every OCL statement is essential. There are total 16 parameters which were discussed and used to evaluate OCL statements. Theoretically, this approach of analyzing each statement helps in recognizing whether OCL statements matches with the medical rules or not. This is generally done by comparing them with the available medical rules for correctness, completeness, conformance, and fault tolerance. Eventually the approach is robust, reusable, and interoperable. The 11 parameters [52] are as follow:

Parameter	Definition	Example
1. Ntraversals	This parameter defines an OCL expression from starting to end. This means if an OCL expression contains attributes of two different class diagram then depending on their back and forth movement the traversal value counter is set.	***self.messageType="k" implies self.surgery=99.*** These two attributes belong to two different classes as shown in figure 4 therefore, traversal value is 1 as it travels from one class to another only once.
2. Ntypes	This parameter gives the total number of data types that are being used in an OCL expression.	***self.basis=57 implies self.surgery=14 or self.surgery=15 or self.surgery=16 or self.surgery=20 or self.surgery=95 or self.surgery=97*** According to the class diagram in figure 4 the data type of basis and surgery is Integer, therefore, the ntypes is 7.
3. Order Type	This parameter based on the total number of different data types gives the order type	***self.messageType='O' implies self.commonFields.basis=57 or***

	Complexity	complexity i.e., it is equal to the highest number of data type in a OCL statement.	***self.commonFields.basis=60 or self.commonFields.basis=70 or self.commonFields.basis=74 or self.commonFields.basis=75 or self.commonFields.basis=76 or self.commonFields.basis=79 or self.commonFields.basis=98 and self.basis=2 implies self.surgery <> 10*** The data types involved in the OCL statement are string, integer, Boolean. The order type complexity is Integer, String, Boolean i.e., Integers (*basis*) are 7, String (*messageType*) is 1 and Boolean (and, <>) 2.
4.	Nclauses Required	The total number of statements starting with "self" required for medical rules.	***self.cancerMessage.messageType= 'D' or self.cancerMessage.messageType= 'O' implies self.surgery=99.*** The required number of clauses for the above statement is 3. This means the medical rule must have 3 clauses for completeness.
5.	Number of Missing Clauses	The number of missing statements starting with "self" for a medical rule. This is achieved by comparing with available rules or done by the medical IT officer.	In the above example we know that 3 clauses are required for the above medical rule. ***self.cancerMessage.messageType= 'D' implies self.surgery=99*** This implies that one clause is

		missing in the above rule.
6. Completeness Constraint Rule	This calculates the completeness of a medical rule based on the difference of the number of total clauses over number of missing clauses. The formula is: $1 - \frac{number\ of\ missing\ clauses}{number\ of\ required\ clauses}$	***self.messageNumber->size()>0*** The above statement states for all the *messageNumber* size must be greater than 1. This statement includes all the *messageNumber*; therefore, completeness constraint value is 1.
7. Completeness Traversals	The completeness of traversal from one class to another class based on OCL statement. The formula is $\frac{(number\ of\ clauses-1)+(\frac{1-number\ of\ traversals\ required}{number\ of\ ntraversals}}{number\ of\ clauses\ required}$.	***self.cancerMessage.messageType= 'D' or self.cancerMessage.messageType= 'O' implies self.surgery=99.*** This statement shows the traversal from *CancerMessages* class to *CommonFields* Class i.e.,1 as shown in figure 4. The completeness traversal is (3-1)+(1-0/1)/3 = 1.
8. Number of Traversals Required	This parameter defines the number of traversals required for an OCL statement.	***self.cancerCaseNumber->size() >0 implies self.cancerMessage.messageNumber->size()> 0*** *cancerCaseNumber* and *messageNumber* belongs to two different classes; therefore, the required traversals for this statement will always be 1.
9. Conformance Iteration	For a given clause in an OCL statement whether it is defined according to the standards in terms of medical rules. The formula is $(completeness\ traversal + conformance\ iteration\ calculation + actual\ conformance)/x *$	**This parameter is explained in detail in section 5.4.**

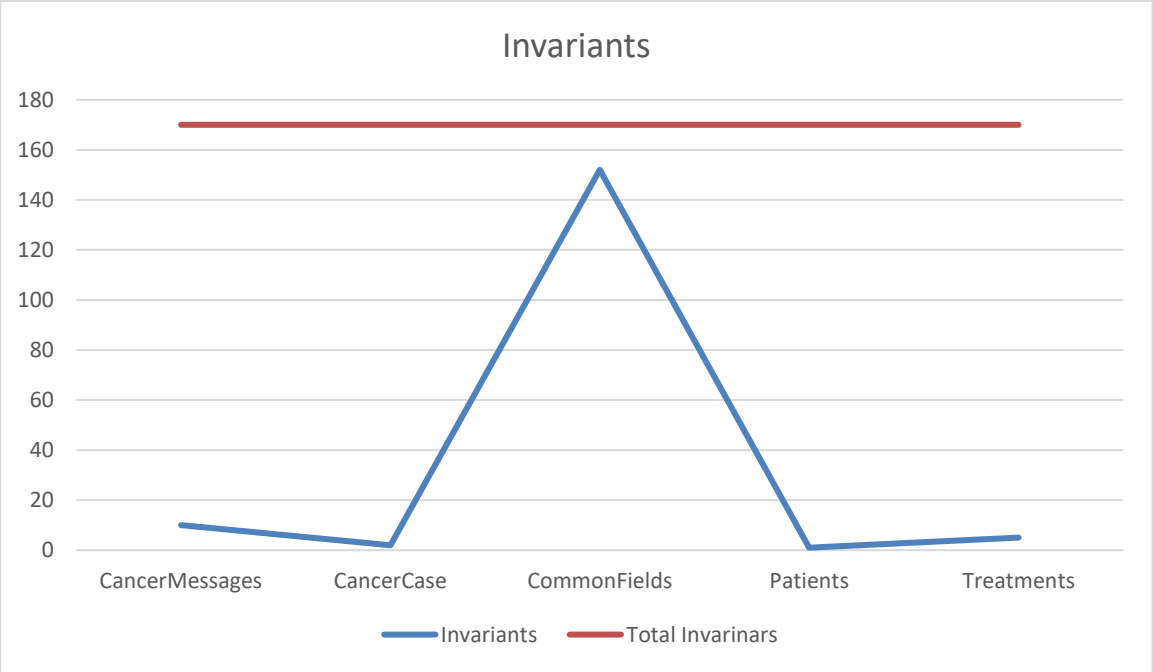
	completeness constraint rule Note: x value depends on the conformance value explained in section 5.4. Its value will be 2 if conformance value is equal to 0 otherwise 3.	
10. Balancing	This is a parameter for capturing the overall model in terms of total number of classes, total number of invariants in each class. Based on these two values we can check the balancing of UML class diagram	**Explained in section 5.1.**
11. Coverage	Based on the completeness constraint rule, we can evaluate whether an OCL statement is fully or partially covered. This is another property of overlapping which defines whether same clauses are used in any OCL statement more than once.	***self.surgery=07 and self.cancerMessage.messageType='H' or self.cancerMessage.messageType='R' implies self.metastase='0' or self.metastase='1' or self.metastase='9' or self.metastase='A'*** ***self.surgery=07 and self.cancerMessage.messageType='H' implies self.metastase='0' or self.metastase='1' or self.metastase='A'*** The above two medical rules are separately defined, but the clauses are same in each OCL statement. Hence, we can merge and make a single OCL statement.

Table 4 Parameters, Definitions, and Examples

Goal is to take these parameters and apply on OCL statements and generate a comprehensive analysis of all medical rules whether these parameters are enough, or some more are required.

5.1 Balancing

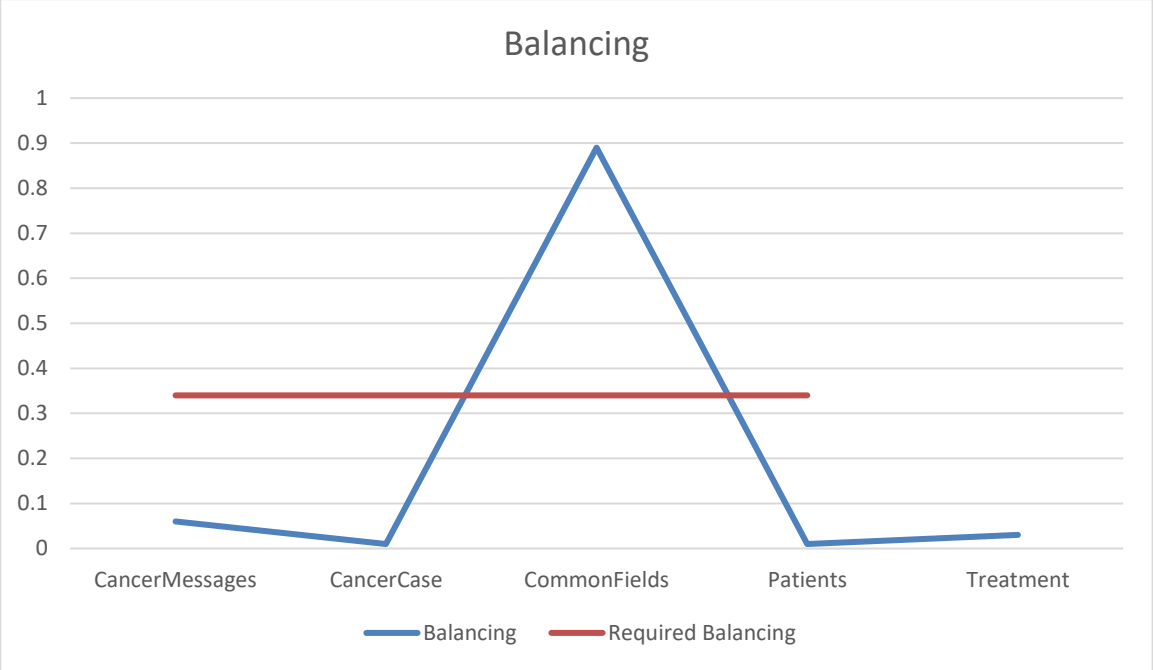
According to the class diagram [6], there are total 5 classes. The total invariants for the 5 classes are 170. The formula to calculate balancing is $\frac{total\ number\ of\ classes}{total\ number\ of\ invariants}$, this gives the required balancing. Now to calculate for each class the equation is $\frac{number\ of\ invraints\ in\ a\ class\ C}{total\ number\ of\ invarinats\ in\ all\ classes}$. The difference of these two values gives us the balancing required for each class.



Graph 1 showing the balancing based on invariants in each class.

Graph 1 illustrates the division of invariants in all the classes. The total number of invariants are 170 and *CommonFields* class contains the maximum number of invariants. X-axis defines the classes namely *CancerMessages*, *CancerCase*, *CommonFields*, *Patients*, *Treatements* and y-axis defines the number of invariants in each class. Red horizontal line points to 170 stating the number of invariants. Blue line shows the different number of invariants in each class. This graph helps in capturing all the invariants required in defining medical rules. *CommonFields* is the main class which has a one to many relations with every other class as

shown in figure 6 therefore, the number of invariants must be more in this class. The graph helps in balancing the model based on whether all the invariants defined for medical rules are implemented or there are some missing which can cause unbalancing between the total number of invariants in the system and overall number of invariants.



Graph 2 shows the balancing of classes based on number of classes over total number of invariants.

Graph 2 describes the balancing of OCL statements in term of invariants defined in each class. Graph 1 gives the number of invariants in each class versus the over all. This graph shows the average of invariants that's needs to be available in each class. To make a model more balanced and structured one needs to use this parameter according to the specifications. In the current scenario completeness, conformance, and correctness are the main area of concern, but parameters selection mainly depends on the domain. The key indication is the point of intersection of balancing value i.e., 0.34 in this case with classes *CancerCase* and *Patients* stating that for this system to be balanced the blue line should be close to this 0.34 value.

The main concern of this study is checking OCL statements for completeness, conformance, and correctness. The rest are the parameters one can use in their studies later. Balancing is mostly important in critical systems where every part of a system needs to have same number of attributes and operations. In my work, the completeness of OCL statements is much

necessary than the balancing of system based on invariants. The main concern is the usage of all the invariants in defining the medical rules for better results.

Below are the two examples giving values using the above parameters.

Example No – 1

OCL Expression

self.messageType='O' implies self.commonFields.basis=57 or self.commonFields.basis=60 or self.commonFields.basis=70 or self.commonFields.basis=74 or self.commonFields.basis=75 or self.commonFields.basis=76 or self.commonFields.basis=79 or self.commonFields.basis=98 and self.basis=2 implies self.surgery <> 10

Parameters	Result
ntraversals	0
Ntypes	Integers (10), String
Order type complexity	Integers (10), String
Number of missing clauses	0
Number of required clauses	11
Completeness constraint rule *	1
Completeness traversal	0
Conformance iteration	0.50
Conformance iteration calculation	0.75
Conformance condition	0.50
X	3

Actual conformance	0.42
Conformance	0

Table 5 Example No.1 OCL Expression

Example No – 2

OCL Expression

self.morfolgi >= 000009 or self.morfolgi >=000019 or self.morfolgi >=000029 and self.DS = 1 or self.DS=7 and self.metastase ='3' implies self.topographi.substring(0,1)='18' or self.topographi.substring(0,1)='25' or self.topographi.substring(0,1)='48' or self.topographi.substring(0,1)='56' or self.topographi.substring(0,1)='71'

Parameters	Result
Ntraversals	0
Ntypes	Real (3), Integer (2), String (6)
Order type complexity	String (6), Real (3), Integer (2)
Number of missing clauses	0
Number of required clauses	11
Completeness constraint rule *	1
Completeness traversal	0
Number of traversals required	1
Conformance iteration	0.50

Conformance iteration calculation	0.95
Conformance condition	1
X	3
Actual conformance	0.65
Conformance	0.33

Table 6 Example No.2 OCL Expression

5.2 Completeness

In the terms of medical rules completeness is calculated by using the following formula [52]:

$$1 - \frac{Number\ of\ Missing\ Clauses}{Number\ of\ Required\ Clauses}$$

The different type of variables is defined as:

int Number of Missing Clauses, Number of Required Clauses;

float completeness;

This formula is rewritten in java language where number of required clauses are automatically calculated from user input (OCL statement). If the user thinks OCL statement lacks some clauses he/she can mention the number of clauses that are missing. This can only be done if the user is already familiar with the medical rules. Based on completeness value, we can relate if a statement if fully correct, partly correct, or written incorrectly. The basic rule of thumb is value 1 is assigned if fully correct, value greater than 0 and less 1 is assigned if partially correct and wrongly written otherwise.

The OCL statement along with the user input and calculated with time stamp is saved in the excel file. This file can be used as a reference later whenever calculating different statements based on some new inclusion of clauses due to new research is Cancer.

Graph 3 Completeness value of 170 medical rules

Graph 3 gives an overview of the completeness rule executed over 170 medical rules. X-axis defines the OCL statements ranging from 1 to 170. Y-axis defines the completeness value of each OCL statement. The rule is simple as stated before in this section value 1 is fully correct, value greater than 0 and less than 1 partially correct and wrongly written otherwise. The graph shows most of the OCL statements are partially correct ranging from 0.50 to 0.80 which is an ideal case. There are very few which are fully correct and often there are statements wrongly written. The key area in the graph is that most of the OCL statements are in the range between 0.5 and 0.9 stating that they are partially correct and hence by improving and comparing medical rules one can improve the completeness property.

5.3 Correctness

Correctness is calculated using the following formula in two steps, firstly calculating the traversals value by [52]

$$\frac{Number\ of\ Missing\ Traversals}{Number\ of\ Required\ Traversals}$$

This tells whether the OCL statement is lacking a class to class relation or not. Secondly it calculates the correctness of that statement. The general principle is the calculation of same OCL statement for completeness and correctness. The formula is

$$\frac{(1 - Completeness\ Traversal)}{Number\ of\ Clauses}$$

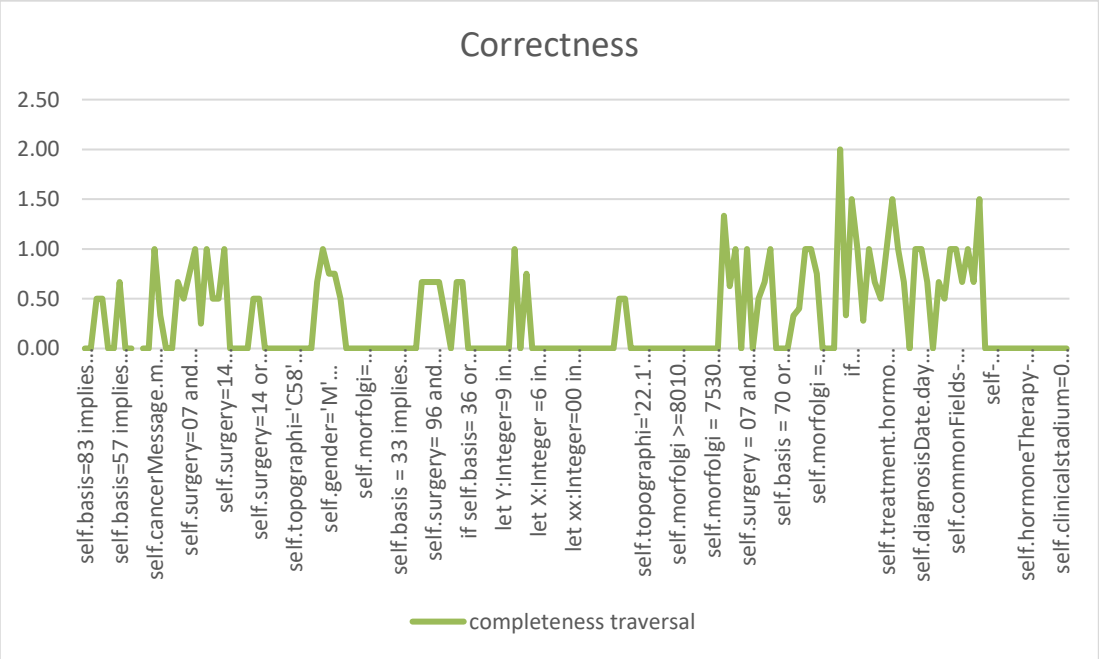
The different type of variables is defined as:

float Completeness Traversal, Correctness;

int Number of Missing Traversals, Number of Required Traversals, Number of Clauses

This formula is converted into java code in which user inputs the OCL statement along with the providing values for Number of Missing Traversals and Required Traversals. The Number of Clauses are automatically calculated by reading OCL statement and counting the number of clauses in it. The rule of thumb is also same for correctness i.e., 1 for fully correct, greater than 0 and less than 1 for partially correctly and wrongly written for all other cases.

This tool calculates a user input to check that if a certain OCL statement is correct or not by providing values and comparing it with medical rules.



Graph 4 Correctness value of 170 medical rules

The graph 4 explains the correctness of each OCL statement having statements on x-axis and correctness on y-axis. The graph illustrates more than half of the statements are partially correct while other are fully correct. There are some statements having correctness value greater than 1 which means OCL statements are either not written correctly or the input values are wrong.

5.4 Conformance

The third parameter measures the conformance value of OCL statement [52]. In software Engineering principles conformance means if a certain product, software is performing according to the specified standards. In this thesis conformance is measured using the values obtained from completeness and correctness i.e., verifying both parameters according to the standards and quality using conformance formulas stated below.

Conformance value is calculated following a series of steps. The formula for calculating conformance contains variables that are required to evaluated first hence the formula is [52]:

$$Completeness\ Constraint\ (Completeness\ Traversals + Conformance\ Iteration + Conformance\ Condition\ )/x$$

To evaluate the correct conformance value of each OCL statement we need to calculate these values first for each statement. The Completeness Traversal is calculated using following 2 steps formulas

$$\frac{Number\ of\ Missing\ Traversals}{Number\ of\ Required\ Traversals}$$

$$\frac{(1 - Completeness\ Traversal)}{Number\ of\ Clasues}$$

This both formulas compute the completeness traversal value for each statement. The data type of all the variables is float. The sole reason for keeping all variables type same is the conformance value will be in decimals.

Conformance Iteration is the value to be evaluated. This evaluates the iteration over class based on OCL constraint is right or not. Hence the formula is

$$\frac{(Number\ of\ Clauses\ -1) + Conformance}{Number\ of\ Clauses}$$

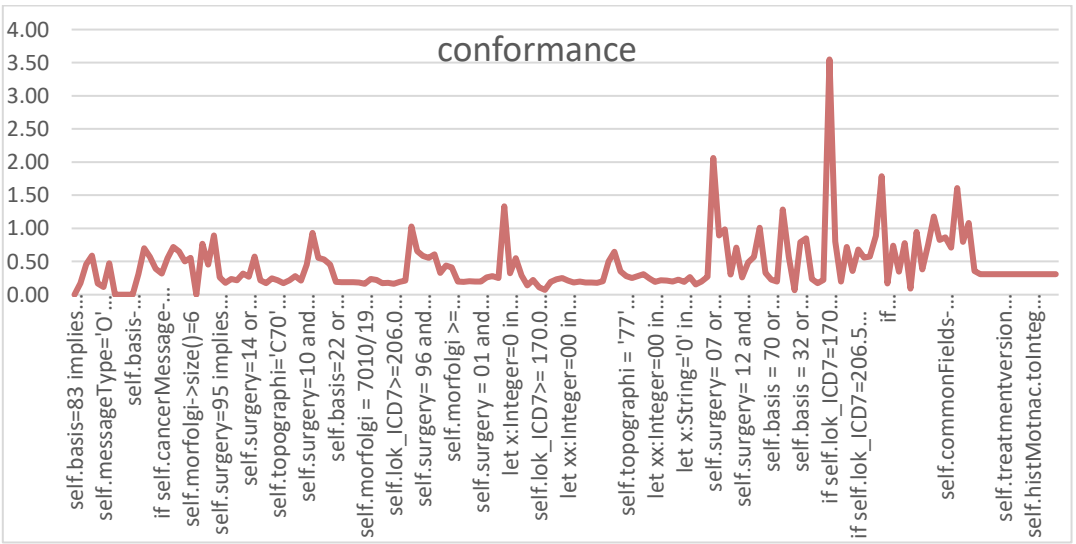
In the above formula Conformance value is not calculated, therefore, this value is calculated using the main conformance formula in a different way i.e.,

$$\frac{Completeness\ Traversal + Conformance\ Iteration + Conformance\ Condition}{x}$$

Conformance iteration starts from 1 to Number of Clauses of an OCL statement iterating over collection of attributes. Variable x value is evaluated based on conformance iteration. X is 2 if conformance iteration is 0 and 3 otherwise. Conformance Condition value can be 1, 0.5, 0 depended on user input. This formula calculates the Conformance value required to calculate the Conformance Iteration.

If we recall the Actual Conformance formula, all the required values are evaluated, and we can easily calculate the Actual Conformance of each OCL statement. 1 for correct, 0.5 for partially correct and wrongly written or calculated statement otherwise.

Tool has a read property which can get each OCL statement on button click from excel file created on basis on CRN medical rules. This helps in checking the completeness, correctness and conformance of all statements comparing it with medical rules.



Graph 5 Conformance value of 170 medical rules

Graph 5 shows each OCL statement versus the conformance value. The graphs show an abrupt behavior in some OCL statements ranging from1.50 to 4.00. This illustrates that those statements are not written according to the standards defined. This could be of different reasons i.e., human error of wrong input values, wrongly written OCL statement, error in calculation etc.

5.5 Summary

This is the main chapter of this thesis which illustrates and evaluates all the parameters that are required for the analysis of automated cancer registry system. This chapter includes the definition with examples of the 11 parameters that are being used to evaluate completeness, correctness, and conformance. The purpose of analyzing cancer registry based on these is the fact that in automated cancer registry medical rules are stored following some verifications and validations, therefore, to check every medical rule is very difficult as one needs to have knowledge of every rule. OCL is used to convert these medical rules and stored them in the cancer registry by calculating the number of attributes used in a medical rule, the conditions and values applied on the medical rules etc. Using completeness, correctness, and conformance formulas to evaluate every medical rule based on OCL constraints helps in determining which medical rules are written and stored correctly and to use those rules as a base for other incoming medical rules.

Chapter 6

6 Tool Support

After the comprehensive understanding of the domain in chapter 3 section 3.2, building a large-scale OCL constraint repository medical rules in chapter 4 and doing an analysis based on the key principles of the created case study in chapter 5. A tool based on the framework in figure 7 to include a repository for all the medical rules is build.

The main perspective of this thesis is to first collect all the necessary domain data along with the understanding of medical rules (written in medical language) and translate them in computer science language. OCL [7] [11] is the computer science language for translation of medical rules by applying OCL constraint using tagging mechanism on the UML class diagrams. There are total nine attributes as stated earlier in chapter 1 that are used for medical rules. All these medical rules are divided into different classes based on these nine attributes as shown in figure 6. The three main rules using the parameters defined in chapter 5 for ensuring the quality of data [2] i.e.,

1. Completeness
2. Correctness
3. Conformance

There are total five main classes containing 170 OCL constraints applied over 52 attributes.

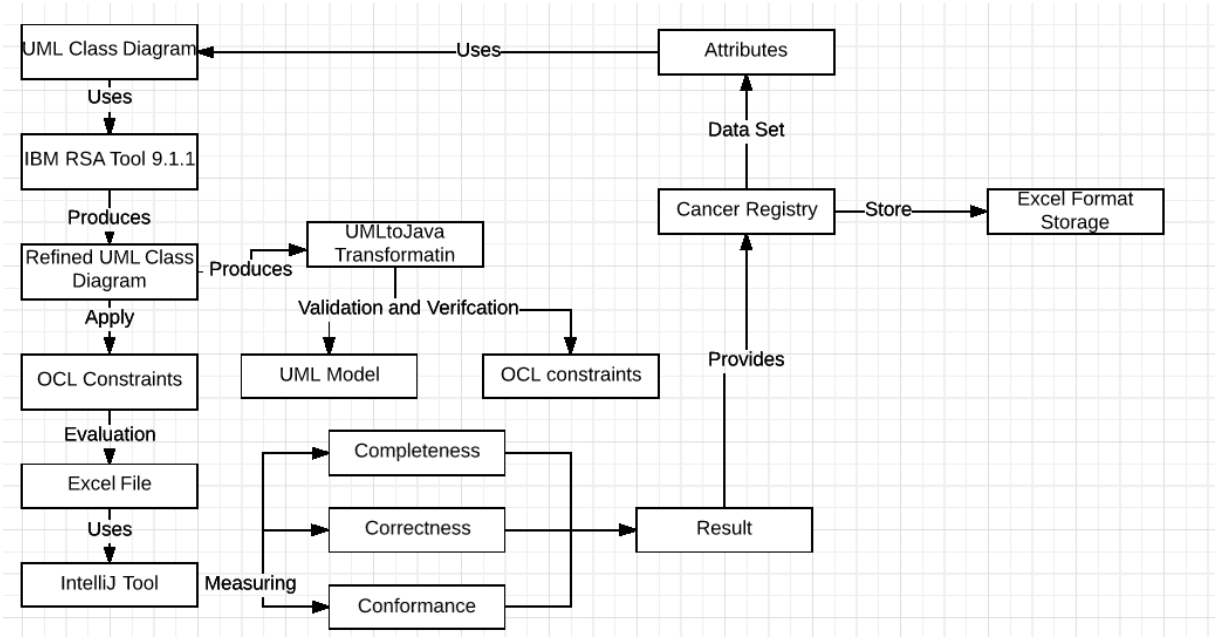


Figure 7 Model-based Tool Analysis Framework for OCL Constraints

A comprehensive analysis of Cancer Registry provides a data set of attributes defined in all the registries. Based on the data set a UML class diagram is created in i.e., figure 6 in chapter 4 section 4.2.3. This class diagram is used by IBM RSA tool for refining and producing a UMLtoJava transformation class diagram for verification and validation of created UML class diagram i.e., figure 5 in chapter 4 section 4.2.3. The evaluation of OCL statements created from medical rules is a two-step process. First step is the conversion of all medical rules using class diagrams to OCL using OCL expressions either directly applied to the class diagram or written first and tested afterwards. If the conversion of all statements is correct i.e., the window shows successfully created than the OCL statements in second step is retrieved from excel file using IntelliJ tool. The tool evaluates the data quality and integrity of each OCL statement based on completeness, correctness, and conformance formulas. The evaluation result of each statement is provided to cancer registry to store it in excel format for later use.

6.1 Prerequisites

Following are the tools, libraries required to build the tool

- IntelliJ IDEA 2017.1
- JavaFx
- Apache-poi.jar 3.16
- Poi-ooxml-3.16.jar
- Poi-3.16.jar

6.1.1 IntelliJ IDEA 2017.1

It is a programming tool [62] rapidly increasing in terms of popularity and usage. The benefits of using IntelliJ are refactoring and plugin stability. If a new plugin is required, it is downloaded automatically when typed. Building a desktop application was a bit easy with this tool because of its shortcuts and easily interaction of different files.

6.1.2 JavaFx

This is an alternative to spring frameworks provided by java [63]. Although it is still in progress, but this is one of the best choice for building desktop applications. It also provides many built-in features required for a desktop application and makes interaction of controllers, classes, and fxml (UI) files easy.

6.1.3 Apache-poi.jar 3.16

The result of OCL statements were to be stored, analyzed, and reused. The selected format was .csv i.e., to interact with excel files apache-poi library [64] is used. This library has rich features using java as programming language and interpreting it with excel workbook. Number of lines of code in java creates a simple workbook containing sheets and records on which any operation can be performed.

6.1.4 Poi-ooxml-3.16.jar and Poi-3.16.jar

These are used for the versions compatibility of excel files. The functions used to interact with excel file before and after year 2003 are different.

For the three evaluation parameters completeness, conformance, and correctness created three separate graphical interfaces for calculation. Parameters specific interfaces are designed. The interfaces are named as completeness.fxml, correctness.fxml, and conformance.fxml respectively.

6.2 Implementation

A desktop application using java and its corresponding libraries is described earlier. Section 6.1 gives an overview of the tool build in IntelliJ. The tool consists of three different interfaces for completeness, correctness, and conformance.

Figure 8 Cancer Registry

This is the main interface of the desktop application. It contains two items in the menu bar namely views and help. Help tab includes the general information regarding the working of the application and defining the parameters in terms of software engineering principles and the importance of them in cancer registry. The view tab includes three menu items namely

completeness, correctness, and conformance. Each item opens in a separate interface having variables, labels, text according to the selected interface.

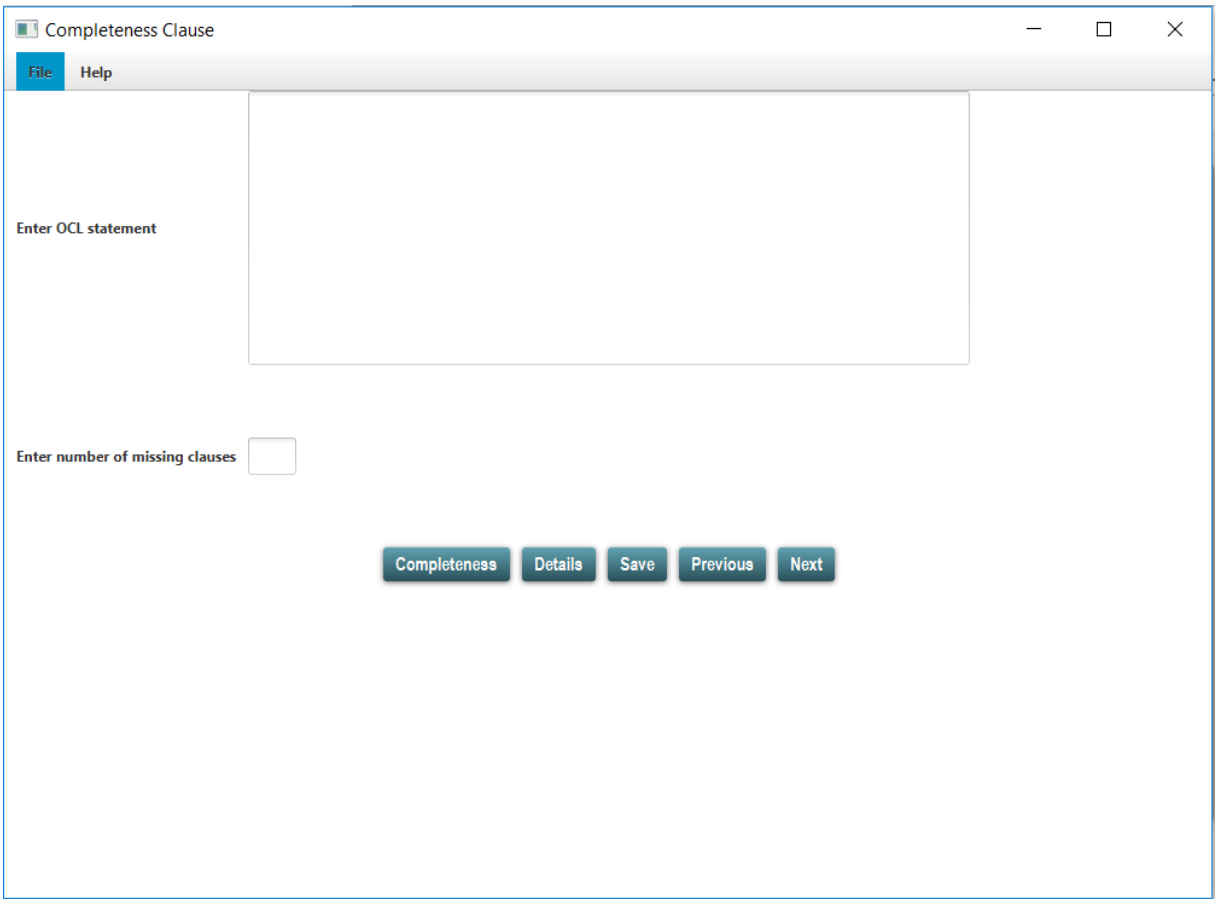


Figure 9 Completeness Interface

After clicking the completeness in the menu item list, it opens the above interface containing the required values to calculate the completeness of each OCL statement. The IBM RSA tool explained in chapter 4 section 4.2.2 helps in creating the OCL statements from the medical rules and python files provided by the Cancer Registry of Norway. The excel file containing all the OCL statements of medical rules is attached with this application using Apache POI library of java which helps in interaction with excel files and performs operations. The interface includes a textarea in which OCL statement is either written or retrieved from the excel file. TextField shows the number of missing clauses in each OCL statement. If user enters a specific value in this field than that value is used for calculation otherwise it is 0 by default. Next and previous button helps in retrieving the row data from excel workbook. Save

button records the detailed answer in another excel file which can be later used for evaluation of OCL statements. Detail button gives an overview of all the variables, values that were used in the calculation of completeness along with some additional information. The completeness button gives the value ranging from 0 to 1. Following example shows an overview of how the interface looks when provided values.

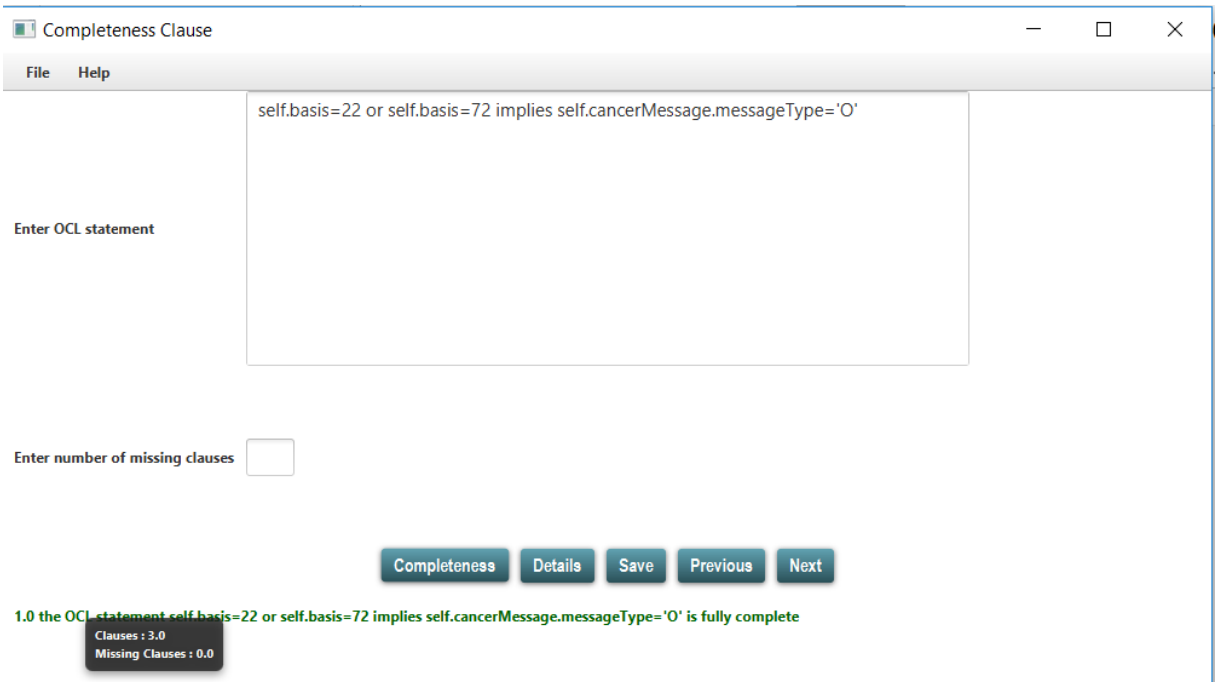


Figure 10 Completeness Calculation

The above figure shows the calculation of OCL statement. This statement is taken by clicking next button and retrieved it from excel workbook, textfield is left empty therefore the default value will be zero. The calculation shows that the provided OCL statement is complete in terms of availability of all clauses required for a certain medical rule. The green label shows that it is fully correct. The black rectangle box is a tool tip showing the number of clauses in the present statement and the number of missing clauses.

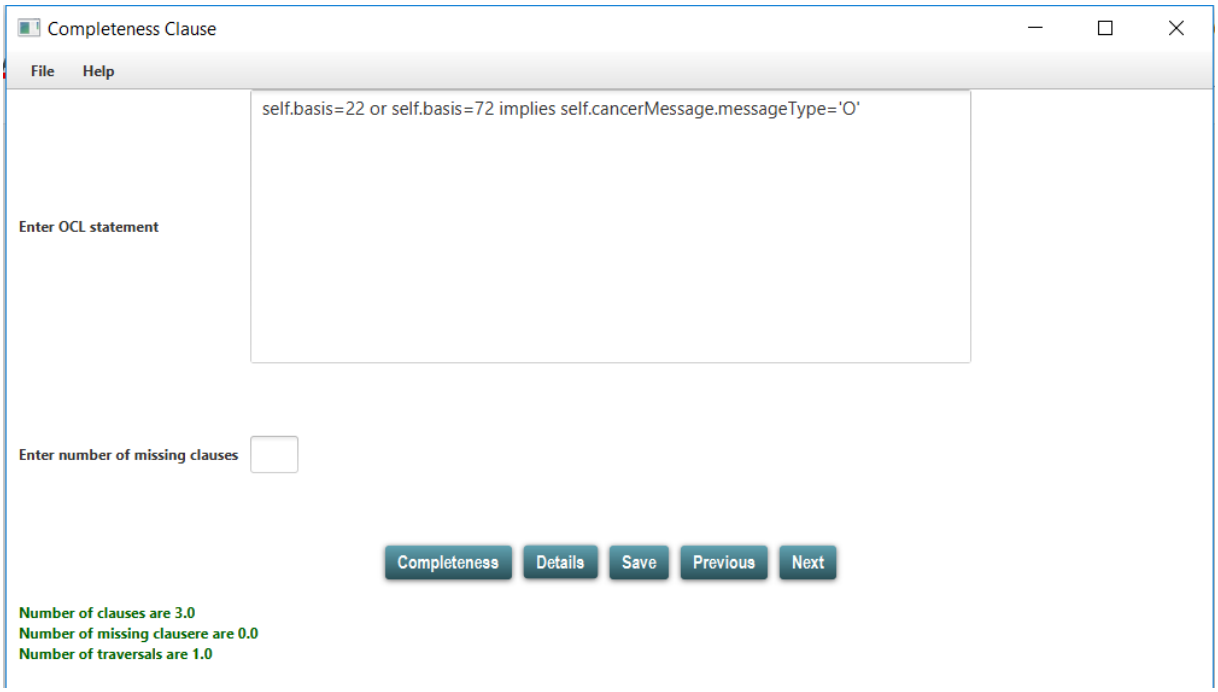


Figure 11 Detail of variables in Completeness

The details button shows the number of traversals value for the above written statement is 1. If we recall the UML class diagram of CRN explained in Chapter 4 section 4.2.2 basis and *messageType* belongs to two different classes namely *CancerMessages* and *CommonFields*. Therefore, the statement shows the execution of OCL statement starts from *CommonFields* class and ends at *CancerMessages* class.

In the same correctness and conformance interfaces works but the variables, labels, values are different in both.

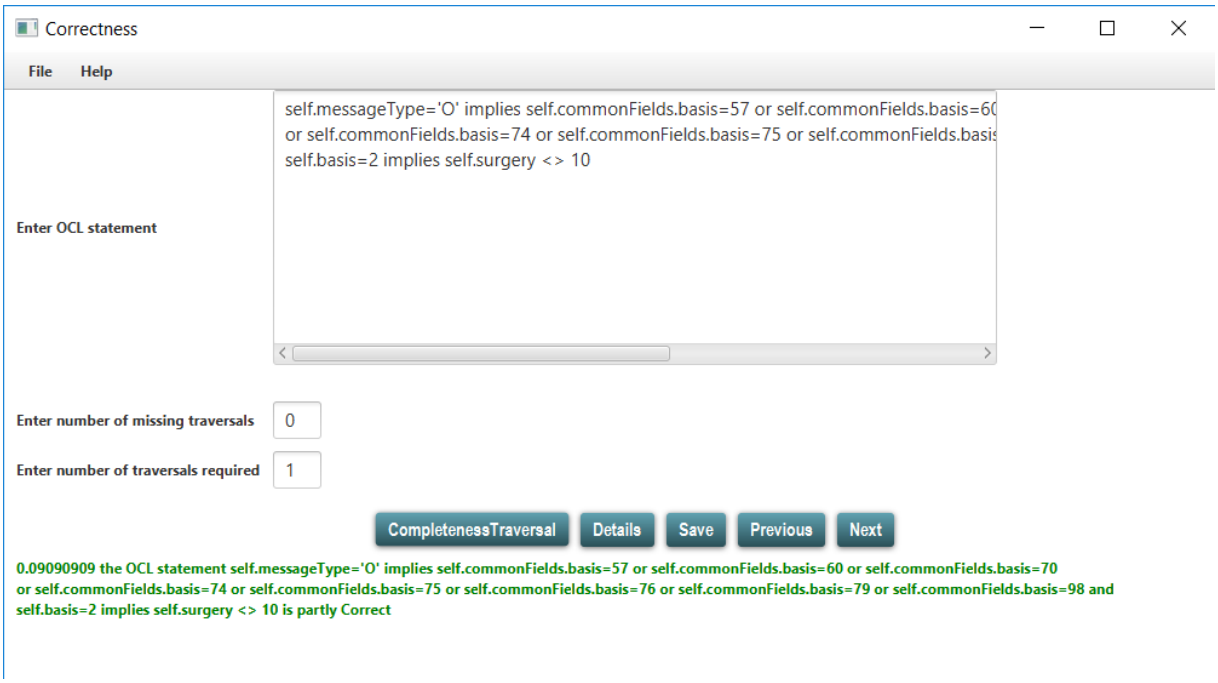


Figure 12 Correctness Calculation

The above figure shows the correctness evaluation of an OCL statement. The formula includes the number of traversals missing and required. As the above statement shows there is only one traversal i.e., from *CancerMessages* class to *CommonFields* class. Therefore, the required traversal in this case is 1.

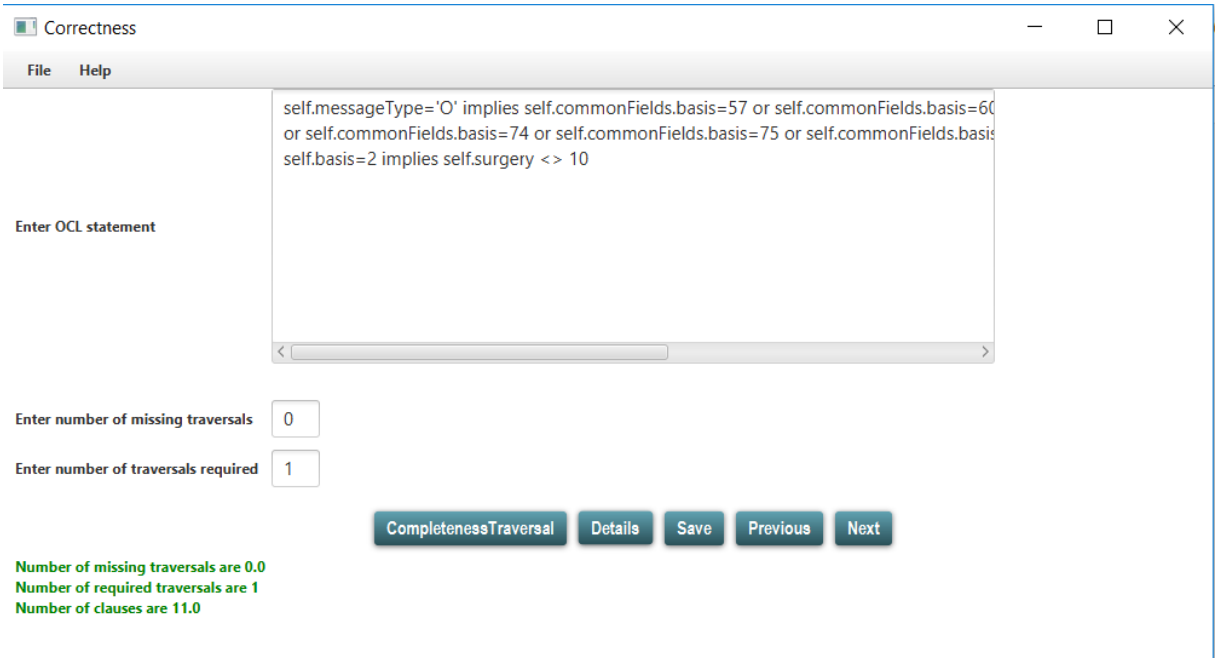


Figure 13 Detail of variables in Correctness

The correctness formula also includes some other variables in its calculation. The details button shows the variable i.e., the total number of clauses in the OCL statement. The next, previous and save button works same as described in the completeness calculation above.

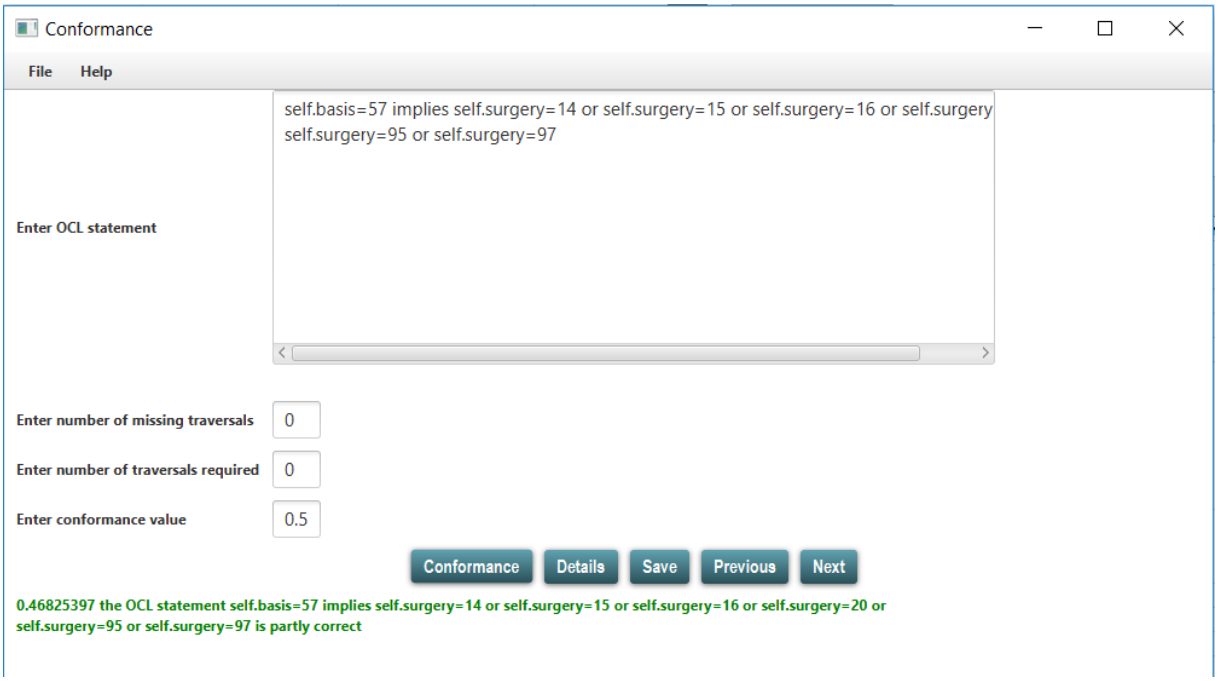


Figure 14 Conformance Calculation

The above figure shows the evaluation of OCL statement based on conformance value. The interface works in the same as for the above two but with different values into the conformance formula. The new variable used in this evaluation is the conformance value which is calculated based on the inner calculation of conformance iteration explained in detailed in Chapter 5 section 5.6. This parameter evaluates correctly if completeness and correctness are calculated properly, because the result of these two parameters are used in the conformance formula and therefore, checks the statement for data integrity and standards i.e., Conformance of OCL statements.

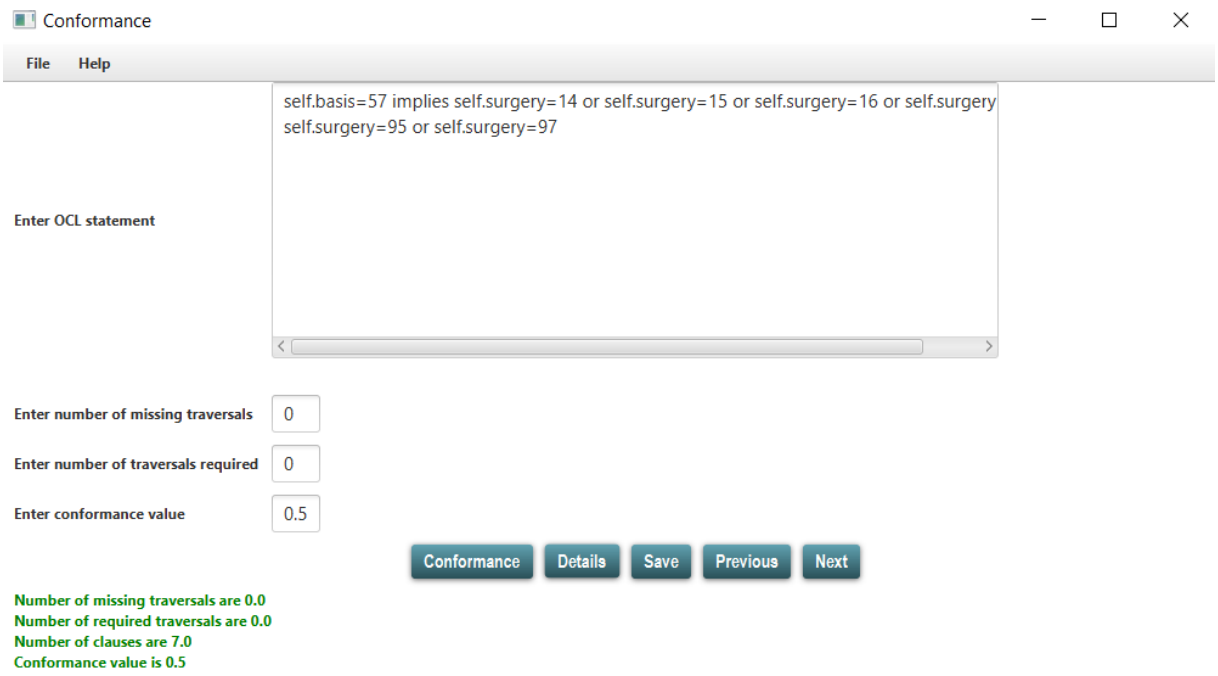


Figure 15 Detail of variables in Conformance

The conformance details button shows the input and the automatically calculated value from the OCL statement. Number of clauses is the variable which is automatically calculated from the OCL statement and is used in the conformance formula for the evaluation. The next, previous, and save button work as explained earlier.

The tool illustrates the evaluation of all OCL statements following a very simple and easy to use interfaces including automatically calculated values and some inputs provide by the user. The tool helps in understanding each OCL statement in terms of completeness and correctness which is the main research goal of this study. Most of the statements evaluated through this tool are mostly partly correct which means there is still much work needed to be done in this area to make these statements from partly correct to fully correct. According to the research and tool calculation, overlapping of OCL statements serves as one of the causes of partly correctness, completeness, and conformance of each OCL statement.

6.3 Summary

This chapter includes a model-based framework according to which a desktop application is build. The framework i.e., figure 7 concludes that by analyzing a cancer registry based on the attributes it uses one can create a UML class diagram and verify and validate the medical rules applied on those attributes using OCL constraints at design level. This chapter includes the prerequisites required for building and understanding of desktop application. Javafx and Apache Poi are the main interface and library that is used in creation and storing of medical rules into excel format. Implementation section briefly explains how the tool works. The tool supports two modes of inputs i.e., either user enter OCL constraint or retrieve it from the excel file which contains medical rules expressed in OCL using IBM RSA tool. The tool calculates completeness, correctness, and conformance of OCL constraint and saves it in an excel format so that it can be used for referring in future.

Chapter 7

7 Conclusion

The framework in figure 7 helps in describing a general model of creating class diagram through domain analysis of a cancer registry highlighting the used attributes using IBM RSA Tool for verification and validation of created OCL constraints. IntelliJ tools helps in evaluating each OCL constraint based on the completeness, correctness, and conformance property. Following paragraphs provides the conclusion obtained using tool and storing it in the excel format.

The graphs in chapter 5 section 5.3, 5.4 and 5.5 shows the evaluated values of all OCL constraints in terms of above completeness, correctness, and conformance. Using OCL constraints on the figure 6 in which class diagram on cancer registry is created helps in evaluating each class along with its attributes efficiently and effectively.

The graph 5.3 gives completeness percentage of OCL constraint in terms of fully correct, partially correct, and wrongly written. The percentage of OCL statements in terms of fully correct is more than fifty percent of all. This means that using the above framework approach helps in making medical rules effective and efficient in terms of completeness. This approach verifies the completeness of OCL statements based on whether a rule defined is complete or not. The rules data set provided by the Researchers of Simula are mostly complete in sense of all the variables required.

The graph in section 5.4 defines the correctness of each OCL statement. This parameter gives accurate value if completeness is calculated correctly. If an OCL statement is wrongly evaluated in terms of completeness, then the correctness will also be wrong. The correctness parameter uses the value obtained from completeness to verify each OCL statement. This means more than 50 percent of the OCL statements are fully correct. This is the ideal case as one cannot have fully correct statements but near to correct statements. The correctness helps in describing the loose ends in the model. The correctness mostly depends on the input output behavior. In software engineering correctness defines the interaction of a user with a system and how the system should operate. Therefore, the correctness value helps in improving the UML model in terms of expected interaction and the behavior of each OCL statement.

The graph in section 5.5 defines the conformance of each OCL statement. This parameter also works well if completeness and correctness are working properly. The reason of using

completeness, correctness and conformance principles is the interaction and interdependence of each other in a system. This checking provides a three way of validation of each OCL statement and helps in improving quality. Conformance graph shows most of the statements are partially correct i.e., in the range of 0.50 mostly. This gives a detailed description of how a statement is following standards. In other words, how much accurate and near to real life is the UML model.

The above evaluation using the framework in figure 7 concludes that the results achieved from these calculations are very helpful in determining the authenticity of medical rules under the defined standards. Most of the values are in the range between 0.5 to 0.9 i.e., graph 5.3, 5.4, 5.5 stating that more than 50 percent of the OCL constraint repository rules are complete, correct and according to the standards.

Bibliography

1. Ray, A. and R. Jetley, *Model-based development: a new approach to engineering medical software.* Biomed Instrum Technol, 2010. **44**(1): p. 51-3.
2. Wang, S., et al., *MBF4CR: A Model-Based Framework for Supporting an Automated Cancer Registry System*, in *Modelling Foundations and Applications: 12th European Conference, ECMFA 2016, Held as Part of STAF 2016, Vienna, Austria, July 6-7, 2016, Proceedings*, A. Wąsowski and H. Lönn, Editors. 2016, Springer International Publishing: Cham. p. 191-204.
3. Siegel, R.L., K.D. Miller, and A. Jemal, *Cancer statistics, 2015.* CA: A Cancer Journal for Clinicians, 2015. **65**(1): p. 5-29.
4. Zachary, I., et al., *Information Management in Cancer Registries: Evaluating the Needs for Cancer Data Collection and Cancer Research.* Online J Public Health Inform, 2015. **7**(2): p. e213.
5. Shuai Wang, H.L., Tao Yue, Shaukat Ali and Jan Nygård, *Domain Analysis in Cancer Registry of Norway (MBE4CR Project).*
6. Shiki, N., et al., *Unified Modeling Language (UML) for hospital-based cancer registration processes.* Asian Pac J Cancer Prev, 2008. **9**(4): p. 789-96.
7. Cabot, J. and M. Gogolla, *Object Constraint Language (OCL): A Definitive Guide*, in *Formal Methods for Model-Driven Engineering: 12th International School on Formal Methods for the Design of Computer, Communication, and Software Systems, SFM 2012, Bertinoro, Italy, June 18-23, 2012. Advanced Lectures*, M. Bernardo, V. Cortellessa, and A. Pierantonio, Editors. 2012, Springer Berlin Heidelberg: Berlin, Heidelberg. p. 58-90.
8. Pohl, K., et al., *Model-Based Engineering of Embedded Systems: The SPES 2020 Methodology*. 2012: Springer Publishing Company, Incorporated. 313.
9. UML. *UML class diagram*. Available from: http://www.uml-diagrams.org/class.html.
10. UML, *UML Class Diagrams.*
11. Richters, M. and M. Gogolla, *On Formalizing the UML Object Constraint Language OCL*, in *Conceptual Modeling – ER '98: 17th International Conference on Conceptual Modeling, Singapore, November 16-19, 1998. Proceedings*, T.-W. Ling, S. Ram, and M. Li Lee, Editors. 1998, Springer Berlin Heidelberg: Berlin, Heidelberg. p. 449-464.
12. Wikipedia. *Model*. Available from: https://en.wikipedia.org/wiki/Model.
13. *Unified Modeling Lnaguage (UML)*. Available from: http://www.uml.org/.
14. Group, O.M. *OMG*. Available from: http://www.omg.org/ocup-2/index.htm.
15. Gogolla, M., *Employing the Object Constraint Language in Model-Based Engineering*, in *Modelling Foundations and Applications: 9th European Conference, ECMFA 2013, Montpellier, France, July 1-5, 2013. Proceedings*, P. Van Gorp, T. Ritter, and L.M. Rose, Editors. 2013, Springer Berlin Heidelberg: Berlin, Heidelberg. p. 1-2.
16. Ali, S., et al., *A Product Line Modeling and Configuration Methodology to Support Model-Based Testing: An Industrial Case Study*, in *Model Driven Engineering Languages and Systems: 15th International Conference, MODELS 2012, Innsbruck, Austria, September 30–October 5, 2012. Proceedings*, R.B. France, et al., Editors. 2012, Springer Berlin Heidelberg: Berlin, Heidelberg. p. 726-742.
17. Vargas, R.T., et al., *The use of UML class diagrams and its effect on code change-proneness*, in *Proceedings of the Second Edition of the International Workshop on Experiences and Empirical Studies in Software Modelling*. 2012, ACM: Innsbruck, Austria. p. 1-6.

18. Arnold, K. and J. Gosling, *The Java programming language*. 3rd ed. The Java series. 2000, Boston: Addison-Wesley. xxiv, 595 p.
19. Leroux, D., M. Nally, and K. Hussey, *Rational software architect: a tool for domain-specific modeling.* IBM Syst. J., 2006. **45**(3): p. 555-568.
20. Budinsky, F., S.A. Brodsky, and E. Merks, *Eclipse Modeling Framework*. 2003: Pearson Education. 512.
21. Weir, H.K., et al., *Evaluation of North American Association of Central Cancer Registries' (NAACCR) data for use in population-based cancer survival studies.* J Natl Cancer Inst Monogr, 2014. **2014**(49): p. 198-209.
22. *Data Set*. 2012; Available from: https://www.cdc.gov/rdc/data/b1/npcr_css_rdc_datadictionary_2011.pdf.
23. (NAACCR), N.A.A.o.C.C.R., *Data Standards and Data Dictionary*.
24. *NAACCR Data Edits*
25. NAACCR. *Data Edits*. Available from: https://training.seer.cancer.gov/operations/standards/scope/edits.html.
26. IBM. *Cancer Registry Model*. Available from: https://www.ibm.com/support/knowledgecenter/en/SS9NBR_9.1.0/com.ibm.ima.using/usi_ibm_im/tut/can_reg_wor/wkd_exp_intro.html.
27. IBM. *Entity Relationship NAACCR*. Available from: https://www.ibm.com/support/knowledgecenter/en/SS9NBR_9.1.0/com.ibm.ima.using/usi_ibm_im/tut/can_reg_wor/relshps.html.
28. CDC. *NPCR*. Available from: https://www.cdc.gov/cancer/npcr/about.htm.
29. CDC, N.a., *National Program of Cancer Registries (NPCR)*.
30. (CDC), C.f.D.C.a.P. *Cancer Registries Amendment Act*. Available from: https://www.cdc.gov/cancer/npcr/amendmentact.htm.
31. *NPCR Software and tools*. Available from: https://www.cdc.gov/cancer/npcr/tools/index.htm.
32. *NPCR-AERRO*. Available from: https://www.cdc.gov/cancer/npcr/informatics/aerro/index.htm.
33. Badar, F., et al., *Epidemiology of cancers in Lahore, Pakistan, 2010-2012: a cross-sectional study.* BMJ Open, 2016. **6**(6): p. e011828.
34. Bhurgri, Y., *Karachi Cancer Registry Data--implications for the National Cancer Control Program of Pakistan.* Asian Pac J Cancer Prev, 2004. **5**(1): p. 77-82.
35. Badar, F. and S. Mahmood, *The State of Cancer Registration in Pakistan.* J Ayub Med Coll Abbottabad, 2015. **27**(2): p. 507-8.
36. Badar, F. and S. Mahmood, *Hospital-based cancer profile at the Shaukat Khanum Memorial Cancer Hospital and Research Centre, Lahore, pakistan.* J Coll Physicians Surg Pak, 2015. **25**(4): p. 259-63.
37. Badar, F., *Cancer registration in Pakistan.* J Coll Physicians Surg Pak, 2013. **23**(8): p. 611-2.
38. Hayat, M.J., et al., *Cancer statistics, trends, and multiple primary cancer analyses from the Surveillance, Epidemiology, and End Results (SEER) Program.* Oncologist, 2007. **12**(1): p. 20-37.
39. Noone, A.M., et al., *Cancer Incidence and Survival Trends by Subtype Using Data from the Surveillance Epidemiology and End Results Program, 1992-2013.* Cancer Epidemiol Biomarkers Prev, 2017. **26**(4): p. 632-641.
40. Institute, N.C. *Geographical Information Systems and Science for Cancer Control*. Available from: https://gis.cancer.gov/atlas/index.php.
41. Yao, N., et al., *Patterns of cancer screening, incidence and treatment disparities in China: protocol for a population-based study.* BMJ Open, 2016. **6**(8): p. e012028.

42. Chen, W., et al., *Cancer statistics in China, 2015.* CA: A Cancer Journal for Clinicians, 2016. **66**(2): p. 115-132.
43. Chen, W.-q., et al., *Report of incidence and mortality in China Cancer Registries, 2008.* Chinese Journal of Cancer Research, 2012. **24**(3): p. 171-180.
44. Chatterjee, S., et al., *Cancer Registration in India - Current Scenario and Future Perspectives.* Asian Pac J Cancer Prev, 2016. **17**(8): p. 3687-96.
45. Takiar, R., D. Nadayil, and A. Nandakumar, *Projections of number of cancer cases in India (2010-2020) by cancer groups.* Asian Pac J Cancer Prev, 2010. **11**(4): p. 1045-9.
46. Storm, H.H., *The Danish Cancer Registry, a self-reporting national cancer registration system with elements of active data collection.* IARC Sci Publ, 1991(95): p. 220-36.
47. Storm, H.H., *Health care system, cancer registration and follow-up of cancer patients in Denmark.* IARC Sci Publ, 1995(132): p. 48-50.
48. Storm, H.H., et al., *The Danish Cancer Registry--history, content, quality and use.* Dan Med Bull, 1997. **44**(5): p. 535-9.
49. CDC. *USCS*. Available from: https://nccd.cdc.gov/uscs/.
50. Chen, W., et al., *National cancer incidence and mortality in China, 2012.* Chin J Cancer Res, 2016. **28**(1): p. 1-11.
51. Jim Hofferkamp, C. *Standards for Completeness, Quality, Analysis, Management, Security and Confidentiality of Data* 2008; Available from: https://20tqtx36s1la18rvn82wcmpn-wpengine.netdna-ssl.com/wp-content/uploads/2016/11/Standards-for-Completeness-Quality-Analysis-Management-Security-and-Confidentiality-of-Data-August-2008PDF.pdf.
52. Yue, T. and S. Ali, *Empirically evaluating OCL and Java for specifying constraints on UML models.* Software & Systems Modeling, 2016. **15**(3): p. 757-781.
53. Parkin, D.M., *The role of cancer registries in cancer control.* International Journal of Clinical Oncology, 2008. **13**(2): p. 102-111.
54. Weir, H.K., et al., *Evaluation of North American Association of Central Cancer Registries' (NAACCR) Data for Use in Population-Based Cancer Survival Studies.* JNCI Monographs, 2014. **2014**(49): p. 198-209.
55. Ma, J. and A. Jemal, *Breast Cancer Statistics*, in *Breast Cancer Metastasis and Drug Resistance: Progress and Prospects*, A. Ahmad, Editor. 2013, Springer New York: New York, NY. p. 1-18.
56. Bowman, S., *Impact of electronic health record systems on information integrity: quality and safety implications.* Perspect Health Inf Manag, 2013. **10**: p. 1c.
57. Hwang, L.J., et al., *SEER and NAACCR data completeness methods: how do they impact data quality in Central Cancer Registries?* J Registry Manag, 2012. **39**(4): p. 185-6.
58. Larsen, I.K., et al., *Data quality at the Cancer Registry of Norway: an overview of comparability, completeness, validity and timeliness.* Eur J Cancer, 2009. **45**(7): p. 1218-31.
59. Zowghi, D. and V. Gervasi, *On the interplay between consistency, completeness, and correctness in requirements evolution.* Information and Software Technology, 2003. **45**(14): p. 993-1009.
60. *Documentation of Attributes in Cancer Registry*. 2015.
61. IBM. *Rational Software Architect* Available from: https://www.ibm.com/developerworks.
62. JetBrains. *IntelliJ IDEA 2017.1*. Available from: https://www.jetbrains.com/idea/.

63. Oracle. *What Is JavaFX?* ; Available from: http://docs.oracle.com/javafx/2/overview/jfxpub-overview.htm.
64. Foundation, T.A.S. *Apache POI - the Java API for Microsoft Documents*. Available from: https://poi.apache.org/.

Appendix A

As described earlier the three main parameters are completeness, correctness, and conformance. Apart from these three parameters there are two more parameters that helps in defining the UML model in more detail and provides necessary feedback for making UML model more reliable and reusable. These parameters are of less importance in this scenario but have a great impact if used in some other practical scenarios. The two parameters as explained in chapter 5 section 5.1 and table 4 respectively are balancing and coverage.

The graphs in section 5.1 show balancing based on total invariants in all OCL statements used and the division of invariants in five classes. The first graph shows the total number of invariants in each class. This calculation helps in determining how the invariants are defined in a UML model. Using this technique, we can re model the UML if the invariants are not defined correctly in each class. Enhancing the limitations of UML model using OCL constraints in terms of invariants helps in identifying and describing all the possible ways of using invariants to provide extensibility and flexibility in the UML model.

The second graph shows to have balanced UML model what number or percentage of invariants should be present in each class. This parameter is not used quite often because it is not possible for all classes to have same number of invariants defined. This parameter still provides enough information about how well a UML model in structured following all the defined standards. This parameter can be of importance in a cancer registry if there are some classes whose ultimate condition is to have same number of invariants defined.

The other parameter coverage helps in understanding the scope of a OCL statement. This means it checks for a medical rule the transformation of value from one to other class is happening or not. This is more of a strategic work in which every OCL statement after comparing with other OCL statements of same type are checked for overlapping as described earlier in section 5.1. This parameter helps in improving the medical rules defined in terms of OCL and try to make a more appropriate medical rule defining all the aspects. It increases the quality and enhances the concept of flexibility and reusability of medical rules.