A Tight Coupling Context-Based Framework for Dataset Discovery

Alaa Alsaig

A Thesis

in

The Department

of

Computer Science and Software Engineering

Presented in Partial Fulfillment of the Requirements

for the Degree of

Doctorate of Philosophy (PhD. of Engineering and Computer Science) at

Concordia University

Montréal, Québec, Canada

May 2023

Abstract

A Tight Coupling Context-Based Framework for Dataset Discovery

Alaa Alsaig, Ph.D.
Concordia University, 2023

Discovering datasets of relevance to meet research goals is at the core of different analysis tasks in order to prove proposed hypothesis and theories. In particular, researchers in Artificial Intelligence (AI) and Machine Learning (ML) research domains where relevant datasets are essential for precise predictions have identified how the absence of methods to discover quality datasets are leading to delay and in many cases failure, of ML projects. Many research reports have brought out the absence of dataset discovery methods that fills the gap between analysis requirements and available datasets, and have given statistics to show how it hinders the process of analysis, with completion rate less than 2%. To the best of our knowledge, removing the above inadequacies remains “an open problem of great importance”. It is in this context that the thesis is making a contribution on context-based tightly coupled framework that will tightly couple dataset providers and data analytics teams. Through this framework, dataset providers publish the metadata descriptions of their datasets and analysts formulate and submit rich queries with goal specifications and quality requirements. The dataset search engine component tightly couples the query specification with metadata specifications datasets through a formal contextualized semantic matching and quality-based ranking and discover all datasets that are relevant to analyst requirements. The thesis gives a proof of concept prototype implementation and reports on its performance and efficiency through a case study.

Acknowledgments

First and foremost, praises and thanks to God, the Almighty, for his showers of blessings throughout my journey to complete my research successfully.

I would like to express my deep and sincere gratitude to my research supervisor, Dr. Vangalur Alagar, for his continuous support throughout the research. Dr. Alagar is and has been always a great supervisor. His supervision is not limited to studies but his kindness and wisdom have enlightened me in valuable aspects of my personal life. Having him believing in me has helped me to achieve and succeed. I am sincerely indebted to him. I am thankful to my co-supervisor Dr. Ormandjieva for her support. Moreover, I thank the members of my examination committee for their comments and criticisms on my initial thesis proposal, which helped me to focus and sharpen my research leading to the contributions in this thesis.

Deepest thanks to my husband Majed Quqandi, who has contained, strengthened and supported me patiently even with the crises that we have been through. Despite having long distant relationship and facing the obstacles to meet each other, he has stayed calm and patient helping to have me focused on my studies and goals. Words are never enough to express my gratefulness and I am sure great and happy days are coming for us.

Genuine thanks to my brother Ammar Alsaig, his wife Sahar Elbilbisi, and his kids who have accompanied and supported me all along the journey. My academic performance would have never been the same without my brother and his continuous cheering. His leads and guides added clarity and value to me and led me to believe in myself. A special thank to my sister in law Sahar for her support and understanding. During hard times she has been a great source of peace and love that made life worthy. Ammar and Sahar have been around at my highs and my lows and words are not enough to express my gratitude to both of them.

Special thanks to my nieces, Wedad and Sarah Alsaig. Who I would love to call my daughters. They have been there to fill my loneliness when my husband had to leave Canada. Seeing their smile everyday added value and meaning to my days. They have been there to share hard and happy times. We faced and struggled a lot, but having our hands held together helped us to stand up after the fail. They have been nothing but the missing piece of love I have ever needed.

Endless thanks to my parents, their sacrifices are infinite. Thanks for their support, prayers, and patience. My parents have been always source of encouragement, trust and love. Their giving and caring cannot be compensated and thanking them is never enough to express my sincere love for them. This love includes all my sisters and brothers who provided all sort of support and surrounded me with the unconditional warmth whenever I needed it.

Special gratitude to my friends I had in this journey. Indeed, *"hard times always reveal true friends"*. During PhD journey you reveal different sides of people who surround you. It is the life lessons that let you become more grateful and devoted to the ones that have provided the minimal favor. As it is hard to exhaustively thank everyone, I say to all people I love, my friends and family, all people who happened to help me even once, thank you.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Humans have gathered, recorded, analyzed, and reused data since the beginning of recorded history. Researchers in many mature disciplines, such as anthropology and astronomy, have been proposing for a long time many theories that required large amount of data, often from other related disciplines, for validating and extending their theories. These probes in turn generated large amount of data. As an example, the research discoveries in the fields of anthropology have depended on the data explored from human biology, evolutionary biology, linguistics, and other social aspects. The field of astronomy is another example in which data gathered from observing the sky over centuries are helping astrophysicists to advance their knowledge on new theories on the origin of universe, and the environmental scientists to study the impact of planetary configurations on climate change. Data was gathered and used by experts in such mature fields in the context of their specific research goals. Such data is called *primary data*, because it is related to the specific analysis goals set for data collection. As opposed to this, data used in a context other than the context of its collection is called *secondary data*. As observed earlier, research in anthropology is dependent on secondary data, the data collected from other disciplines. Hence, gathering primary data, and mixing it with secondary data for analysis based on traditional statistical methods are not new.

In modern times, due to the advent of modern technology, such as internet, distributed systems and web services, large amount of data is generated by a variety of sources that may be interconnected either locally or through the internet. Most of the data from these sources are only secondary data, and may be fused with proprietary (private) data such as internal databases and legacy systems. As an example, in healthcare domain primary data includes genomics data, biological data, clinical data, drug-patient interaction information, and personal preferences of patients in the use of inter-connected (embedded) medical devices, while secondary data may include data from clinical research, patient-centric privacy policies, ethical and legal issues, and environmental data (where the patient lives or lived). The important challenges to overcome in collecting such data are *data volume*, *data heterogeneity*, *data asynchrony*, and *data quality*. Data volume and the frequency of flow cannot be predicted at some data generation mediums. Data heterogeneity refers to non-uniform data format (structured or unstructured), non-conformance of data type, and the data semantics that might cut across many (related) domains. Data asynchrony refers to the rate at which data is generated, updated, and delivered. Quality assessment is challenging, because of the variety of quality attributes ranging from data incompleteness to data coherence (Strozyna et al., 2018). As an example, in healthcare domain in which online offering of personalized healthcare is now regarded as more efficient because it can be tailored to different contexts, data quality, data heterogeneity and data asynchrony are bigger challenges to overcome. Given this multi-faceted challenge, it was realized that the traditional "data management and analysis" methods cannot be adopted anymore, and a new approach is necessary to manage modern data explosion. This awareness lead to Big Data (BD) study, with the great expectation that a new set of techniques and tools in this study will benefit data source providers and data analytics teams.

It is well acknowledged (Dong, Saha, & Srivastava, 2012; Joshi, 2017; Kornegay & Segal, 2013) that BD is collected and managed by organizations for the sole purpose of explorative analysis and deep data analytics, hoping that will lead to "better" management decisions. Data analytics is the process of developing new techniques and tools for data mining and knowledge extraction, inference and prediction. However, it is remarked (Wetherill, 2016) that "only 2% of business executives acknowledge that some positive impact has been achieved through data analytics". From a critical analysis of many papers in BD literature, this lack of success can be attributed to two reasons.

- *Lack of Contextual Compatibility:* The report (NRC, 2013), developed by the National Academy of Sciences, observes that one of the primary reasons for lack of successful analytics is "data collected according to certain criterion (in one context) is often used for data analytics in another context that is not compatible with the original context". The report insists that knowing quality features of datasets at the context of data collection will lead to "better" discrimination in choosing data sources that are relevant to the context of analysis goals.

- *Gap: Lack of Tight Coupling Mechanism:* Researchers in AI (Paleyes, Urma, & Lawrence, 2022; Roh, Heo, & Whang, 2021), ML (Ng, 2021; Paleyes et al., 2022; Roh et al., 2021), and other domains (Koesten et al., 2020; Strozyna et al., 2018; Takeuchi & Yamamoto, 2020) who are intensively engaged in data analytics have now recognized the "gap" that exists between data source providers and dataset analytics teams. The data analytics team can formulate a set of requirements on the data they desire in order to conduct their analysis. This formulation defines the "specific set of requirements" of data analytics. In order to select datasets from open data sources that "match" this set of requirements, it is necessary that the data source providers publish precisely and comprehensively the "metadata" information of the datasets. This is necessary for enhancing the accuracy and relevance of dataset selection to meet the analysis goals. The absence of such a facility at the source end is called the "gap". With such a gap, tools and techniques of analytics, however sharp they are, will only lead to incomplete, inaccurate, and unwanted analysis results.

As of now, removing the above inadequacies remains "an open problem of great importance". In this thesis we propose the *Tight Coupling Framework (TCF)* which is a component-based *contextual framework* that will establish a *tight-coupling* between dataset provider level and data analytics level. Through this framework, the identification and retrieval of datasets that are relevant to the goals of analysis team are achieved in a formal manner.

In 2019, the initial proposal of this thesis titled " Rigorous Approach to Modeling, Management, and Analysis of Healthcare Big Data" first identified the significance of this problem, and proposed a 3-tier architectural framework to realize it. While the work on this proposal has been progressing, four research papers (Bogatu, Fernandes, Paton, & Konstantinou, 2020; Castelo et al., 2021; Chapman et al., 2020; Noy, Brickley, & Burgess, 2019) reporting different approaches to solve this problem have been published. We want to emphasize that compared to them, the research initially proposed in 2019 thesis proposal is the first to highlight the significance of this “open problem”. As we will justify in Chapter 2, in spite of the recently proposed methods the problem still remains “open”, and hence this thesis has distinct merits.

1.1 Data Sources of Interest and Relevance

The data sources from where the datasets are obtained at one end, and the notion of relevance for analysis at the other end define “the context of interest” for dataset discovery. In this section we make clear the types of data sources that are of interest for our study, and make precise the meaning of relevance.

The data sources have been classified in (Strozyna et al., 2018) as follows:

- *Open data sources* (O): This category includes all websites that all internet users can access.
- *Open data sources with registration* (OR): These are websites that provide data only to authorized users.
- *Data sources with partial paid access* (PPA): These are websites that provide a wider scope of information selection after a fee is paid.
- *Paid data sources* (PA): A regular subscription is required for access to these websites.
- *Private data sources* (PR): The information is privately held by an organization for its internal use.

In the context of maritime domain research (Strozyna et al., 2018), it is observed that we can only focus on *quality-based selection of (O) and (OR)* because other data sources are either expected to contain only high-quality data or mostly primary data. We follow this guidance in this thesis, and stick to O and OR data source types.

According to an extensive study done by several researchers (Koesten et al., 2020; Pipino, Lee, & Wang, 2002; Strozyna et al., 2018), the meaning of data quality, although very important for data source selection, has not been completely agreed upon by the data analysis groups in business, technical, and medical domains, and as well as in information sciences and software engineering communities. European Commission (ESS, 2014) has listed a set of quality attributes as standard for web page selection in healthcare domain. In information systems literature (Pipino et al., 2002), quality of data is explained as a "multi-dimensional construct". We pick the quality attributes from Eppler (Eppler, 2006) who originally proposed 70 attributes, and later narrowed it to a set of 16 most important attributes that include *precision*, *completeness*, *consistency*, and *timeliness*. The notion of "relevance" has been researched by many, of which the study of Blum (Blum & Langley, 2018) in the context of Machine Learning (ML) is more exhaustive. We choose the interpretation that "quality" and "relevant to a set of requirements of analyst goals" are "equivalent". Because an analyst is free to include quality attributes in the set of requirements, in this thesis "relevance" includes quality of data. This is essential for data analysts because the quality of data they will use defines the quality of their research outcomes.

1.2 Significance of Thesis

An extensive literature survey reveals that datasets selection problem, in spite of its criticality for BD analytics, has not been adequately addressed until recently. Many analytics experts (Blum & Langley, 2018; Ng, 2021; Paleyes et al., 2022; Polyzotis, Roy, Whang, & Zinkevich, 2018; Takeuchi & Yamamoto, 2020) have pointed out that discovery of quality datasets is made difficult by the current "management practices of data sources". Recent analysis (Gumbrecht & Fox, 2020; Joshi, 2017) of Covid-19 healthcare datasets indicate that a misinterpretation of the results of Covid-19 medication side-effect from the available data could lead again to another global pandemic. It is clear that if datasets that are relevant to specific analysis goals are not presented to the analytics team, either there is a huge overhead for the analytics team to "clean up" the datasets from available data sources or face failure in achieving complete and correct analysis even when the tools and methods used by the team are robust. Motivated by the criticality of the problem, and recognizing the current gap between "BD management" and "BD analytics" within which the dataset selection problem lies, in this thesis we propose a component-based contextual framework to enforce the "tight-coupling".

This framework has three components. At the data source provider end is one component in which every data source provider announces the “metadata” information formally in a directory. The directory structure is rich enough to include information on the context of data collected at the source, the domains, sub-domains, and attributes of collected data, and the quality features and legal/security aspects governing the intended use of data. At the analyst end is one component through which the analysis requirements are formally represented. This will include domain of research, sub-domains, attributes, and quality measures of datasets required. included in the set of requirements. When every data source provider announces the datasets context in a common directory (one component) and the analyst team announces formally the context of dataset requirements for analysis (one component), the third component in our design acts as the *tight-coupling interface* between the two components. The functionality of this component includes browsing the directory, matching the contexts, selecting the matching data sources, and finally applying user-centric semantic-based similarity calculation to filter from the initially selected datasets only those that “best matches” the quality requirements of the analyst. That is, it is a rich “dataset search engine”. These filtered datasets are then ranked by this component and delivered to analyst-level component for further use.

The proposed component structure, namely its sub-components and their interfaces, will be built on firmly grounded software engineering models and theory. Our design is expected to promote less conflicting research outcomes and higher quality and trusted results. What’s more, it will minimize the intentional and unintentional misinterpretation of data because of regulated contextual match making during the initial selection and matching stages. We will formally justify later in the thesis that our approach is both *sound* and *complete*. Soundness means “no junk”, and completeness means that “everything that is relevant” will be delivered. To the best of our knowledge, up to the time writing up this thesis, there is no complete framework that performs this process contextually and formally.

1.3 Contribution of Thesis

Abstractly viewed, the framework design establishes a contextually governed medium for dataset collection to meet the analyst goals. The *Dataset Context* (DSC) is provided by the data source provider. The *Data Requirement Context* (DRC) is provided by the analytics team. They are contextually regulated in the data source selection process. The main tasks performed on contexts in the selection process are as follows:

- *DSC Formation:* DSC is formalized as a rich service model that includes description to attributes, features, data quality, metrics, ontology concept terms, and other contractual information of the data source.

- *DRC Formation:* DRC is formed with respect to the goal of the analytics team. The requirements of analysis are incrementally fetched using a well-established goal-oriented approach referred to as 'Goal, Question, Metric' (GQM) (Basili, 1992; Basili & Weiss, 1984; V. Solingen, Basili, Caldiera, & Rombach, 2002).

- *DSC and DRC Matching:* This task is non-trivial because different types of matching, such as *exact matching*, *partial matching*, and *best matching* are possible. We use "exact match" for concept terms, and "best match" for quality attributes. That is "best" means "what is best for the analyst", using analyst's semantic constraints for "best". We use similarity measures to calculate the degree of "best match" on quality aspects between DSC and DRC. Datasets that have semantic match are ordered according to decreasing similarity scores calculated for quality aspects.

- *Data Source Selection:* The selection of datasets from the ranked collection is "contextual and user-centric". The analytics team gets the full right to review, filter, re-order the selection procedure that best suits the contextual constraints at their end. This process may be iteratively continued until the analytics team gets to select the most relevant datasets.

The above abstraction is realized in many chapters of this thesis, as enumerated below.

- A critical review of dataset selection methods, related specifically to the design/development of dataset engines, is given in Chapter 2. Papers cited above and a few others that have contributed to the current state of the art in dataset discovery and data analysis are reviewed. This critical analysis helps to highlight the significance of our work.

- In Chapter 3 we present an abstract view of the TCF component-based architectural design. This architecture is a composition of the three components, *Provider Component* (PrC), *Analyst Component* (AnC), and *Coupling Engine Component* (CeC). After providing a general overview of the components and their interactive behavior, we list the merits of our design which can be the basis for comparing our dataset search engine with the existing dataset engines.

- In Chapter 4 we discuss the design and implementation of PrC component. This chapter starts with a discussion of the expert support given by OnC component in the creation of metadata of the dataset model. Then it discusses the directory structure in which metadata descriptions are published. The rich user interface for service providers to create metadata descriptions, and other interfaces through which PrC communicates with other design components are explained. A prototype example is given to illustrate the implementation of this component.

- In Chapter 5 we explain the functionalities of AnC component. The chapter first discusses GQM (Basili, 1992; Basili & Weiss, 1984; V. Solingen et al., 2002) approach. The User Interface (AUI) based on GQM is described next. Using the AUI, an analyst inputs the requirements for dataset discovery. This includes, domain and subdomain information, attributes, quality requirements (quality attributes, their weights, and preferences). This component, after interactively processing user requirements, constructs the query vector, and sends to CeC. When CeC responds with a ranked list of metadata descriptions, AnC will apply contextual constraints on the received list, and select the datasets that are suitable for the analyst's context.

- In Chapter 6 we discuss the functionality and features of CeC, the dataset search engine. There are two phases to the search engine. In the first phase, all metadata descriptions that semantically match the semantic part of the query are selected. We use Formal Concept Analysis (FCA) (Alsaig, Mohammad, & Alsaig, 2015; Ghaemi, 2011; Wille, 2009; Wolff, 1988) for semantic matching. In the second phase, the quality features of each metadata description in the selected set is compared with the quality features specified in the query. The amount of closeness (similarity) between them is formally measured. Using the measures calculated for metadata descriptions, we rank them in decreasing order of similarity measure.

- In Chapter 7, first we provide a theoretical complexity analysis of the dataset search engine of CeC, and then a case study to illustrate our approach for dataset discovery. We first validate our prototype implementation with a small number of datasets. Next, we execute the same case study example on a large number of datasets and for different query types. By varying the number of functional and non-functional attributes in metadata descriptions, and varying query types we estimate the run-time performance and comment on the results.

- We conclude the thesis in Chapter 8, with an explanation on how we met our goals, how well we met them when compared with the existing dataset discovery engines, and give a plan of future work that will overcome some limitations on existing design and enhance the performance our approach on large real-world datasets.

Chapter 2

Related Work

In this chapter we restrict our review to papers published on the topics "experience report from AI & ML (Machine Learning) community on the current status of finding quality Datasets", "data quality (DQ) dimensions and DQ assessment methods", "experience report on quality criteria for Dataset selection", "Dataset selection methods", "recent works on Dataset Search Engines", and "attribute and schema based matching and ranking of relevant datasets". We categorize the published papers, as much as possible, in chronological order. We summarize the contributions of papers in each category and comment on their influence in our current work. This way we bring out how the progress and evolution of ideas over a period of 20 years has motivated the new results proposed in this thesis. We add a note of caution that in all publications before 2018 the terms "data source" and "dataset" have been "mixed up", often authors using one term where the other term is meant. Also, the terms "DQ attributes" and "DQ dimensions" are used interchangeably in literature. In the review of papers in this time period we use their terms as it appears in their papers. At the end of this chapter we give a relative comparison of the above papers and explain the set of quality attributes selected for inclusion in the thesis. A critical evaluation and comparison of the two published dataset search engines together with other related work discussed in this chapter with the approach proposed in this thesis appear in Chapter 8.

2.1 Experience Reports from AI & ML Community

It is well known that AI & ML research community is one of the largest group of researchers who pursue data mining and knowledge discovery in many important application domains of national importance. They acknowledge (Paleyes et al., 2022; Polyzotis et al., 2018; Takeuchi & Yamamoto, 2020) that the following three issues are the most important Data Management Steps to be resolved before ML activities can begin in the ML life cycle model.

- Finding data source and understanding their structure is a major difficulty because of data dispersion.
- There is a pressing need for accurate and scalable data collection techniques, especially for emerging new application domains such as "automation of smart factories", which require large amounts of reliable data.
- One of the nine reasons for ML projects to fail is "not having suitable data". They emphasize that the accuracy of machine learning model is intimately tied to the quality of data that it is trained on. They have found from their experience that training on contextually incompatible and corrupted data will "typically lead to reduce model accuracy", and this in turn affects accuracy of prediction built on trained models.

According to ALGION report (ALEGION, 2019), roughly 96% encounter challenges with training data quality and quantity, about 78% of ML projects stall at some stage before deployment, and about 71% of companies "outsource ML activities on data collection, and model development". Consequently, "only 2% of business executives acknowledge (Wetherill, 2016) that some positive impact has been achieved through data analytics". It is remarked (Ng, 2021) that currently 80% of ML workers do "data cleaning" because current datasets do not meet their expectations. During the machine's training period, the context for its successful operation should be coded in the learning algorithm's function (Jain, 2015), however the context (metadata information) on most of the currently available datasets cannot be used easily because they are not well-structured or only informally stated. Thus research in AI & ML community has swayed away "from sharpening code and tools to focusing on improving quality of data", which is hard, time consuming, and thus delays their project completion. They are looking for data scientists to devise accurate and scalable data collection techniques, formally indexed metadata information, and provide easy to understand and usable methods to determine whether a dataset is suitable for a machine learning task (Paleyes et al., 2022; Roh et al., 2021). In Ashmore, Calinescu, and Paterson (2015) the authors identify *relevance*, *completeness*, *balance*, and *accuracy* as the four essential quality properties of datasets for ML research. They explain these terms with reference to ML research and suggest methods for the Data Management team to achieve them.

2.2 Data Quality Dimensions and Data Quality Assessment

The earliest papers Batini, Cappiello, Francalanci, and Maurino (2009); Eppler (2006); Pipino et al. (2002) have extensively discussed the importance of data quality for developing large information management systems. In common they have identified 16 DQ dimensions. These are *Accessibility*, *Volume*, *Believability*, *Completeness*, *Concise Representation*, *Consistent Representation*, *Ease of Manipulation*, *Free-of-Error*, *Interoperability*, *Objectivity*, *Relevancy*, *Reputation*, *Security*, *Timeliness*, *Understandability*, and *Value-Added*. Many years later, the BD research community has made use of one or more subsets of the 16 proposed DQ dimensions in their experimental analysis. From an exhaustive search of published papers related to "Criteria for Dataset Discovery" in specific and in general domains, we selected few papers whose works use DQ dimensions for dataset selection. Among those, the first work to recognize the "data quality" aspect for data source selection was published in 2000 (Mihaila, Raschid, & Vidal, 2000). Almost two decades later the paper (Strozyna et al., 2018) discussed "quality-based selection of datasets" in Maritime domain. Because these two papers are important seminal works in their own right in exploring "dataset search engine approaches" we review them separately in Section 2.3, and in Section 2.4. In the previous section we have summarized the experience reports of AI & ML group of researchers who have identified a small set of quality attributes they want the datasets to satisfy for their research. In this section we review two recent papers. The paper (Koesten et al., 2020) provides a list of DQ dimensions collected from experiments on a large collection of open data sources. The paper (Bian et al., 2020) gives the list of DQ dimensions and DQ assessment methods extracted from the research reports of experts in clinical research domain.

2.2.1 Experience Report-Dataset Practitioners (Koesten et al., 2020)

Two studies were conducted in (Koesten et al., 2020). In Study 1, 69 participants (mostly students) created 269 *data-search diaries* and these diaries were used to understand "what data attributes do people consider when determining the relevance, usability, and quality of a dataset?". A data-search diary is a "user created record of the data search process followed by the use". In Study 2, 360 data summaries created by 80 participants for 25 datasets were analyzed to determine "what attributes do people choose in writing dataset summaries for others".The paper gives suggestions on "how to go to metadata description from their summary", however does not provide a standard template. So, we ignore Study 2, and give below an account of Study 1.

In Study 1, the authors group the data attributes emerging from the data-search diaries into the three high-level themes *relevance*, *usability*, amend *quality.* These are explained below.

1. Relevance refers to how applicable dataset content is to the user task. According to the study, the elements to measure relevance are listed below, annotated with the additional elements we intend to use in the thesis are listed below.

- *Scope:* The scope of data can determined by geographical area or specific time frame. *We will add domain and sub-domain information to help analysts select datasets more precisely.*

- *Granularity:* It refers to the level of detail in a dataset. Some participants considered statistical information such as average, and mean, and others considered specific column headers such as street, city as "level of detail". *In the thesis, statistical measures can be included by dataset providers in the metadata directory. Analysts can view them before making a selection.*

- *Context:* The paper considers the "purpose of data" as an element of data context. *We include the purpose element as one of the "dimensions" of context. The thesis explains "data source context", whose elements are inherited by the datasets provided by that data source.*

- *Documentation:* The participants in the study emphasized that "good documentation" is needed for understandability of data variables, attributes description and data summaries. But, they have not proposed a documentation style/template. *In some studies Bian et al. (2020); Pipino et al. (2002), understandability refers to the level of data comprehension for selection. In BD community Li et al. (2016) there is no "consensus" on what "understandability" means, although many recent works on dataset search engine have used "metadata" descriptions as promoting understandability. We concur with this point of view and provide a formal directory for metadata descriptions which data source providers can use to fill metadata information.*

2. Usability refers to a quality feature to determine the value of a dataset to a certain task. The participants in the study (Koesten et al., 2020) have recommended the following factors to determine the level of usability of a dataset to a certain task.

- *Format:* This refers to features at schema level, such as datatype and other structure information that can contribute to "schema join and integration process". *However, we believe "format" may also refer to the "format of the data provided at the dataset" such as XML and CVS files.*

- *Documentation:* A good documentation promotes understandability of variables and dataset samples. *In the thesis, both "format" and "attribute description" are considered as important features for metadata description. The primary purpose of metadata part is to improve understandability, and hence improve the proper use of data.*

- *Comparability:* It refers to the ability to measure the differences between datasets and how these differences can impact any further processes . This implies that units of measurement has to be defined to compare datasets on different aspects such as semantics of attribute names (column headers), and quality features. *We concur with (Castelo et al., 2021) that current metadata of datasets do not correctly describe datasets and are not rich enough in attribute descriptions and data summaries elements to enable datasets comparison.*

- *References:* It refers to "reference" to the original data source. *The standard of dataset metadata* `schema.org` *includes this feature, however not all datasets include* *the original data source information as some of them are collected manually through surveys. In this thesis, the dataset provider is required to provide this important information in the metadata description..*

- *Access:* The study found that only a small percentage of dataset seekers preferred this feature, because "it is related to license element or other conditions to authorize accessibility to dataset" in this study. *However, accessibility to data sources is an important issue in ML research because they require large amount of data for their analysis. In this thesis we are assuming datasets to come mostly open data sources and from some where registration and authorizations are essential.*

- *Size:* it is defined as the extent of a dataset volume that matches the task. *This is an essential element of importance in ML research. In the original proposal of this thesis, dataset context model, which is published Alsaig, Alagar, and Ormandjieva (2018b), includes the data volume as an element. This is because this element is highly related to analyst needs. Hence, it plays a role in ranking processes that is based on analysts preferences.*

- *Language:* It refers to the language used in the dataset at least in the column headers to help dataset requesters to easily decide the usability of this dataset to their task. *This is an important feature we consider in the metadata of dataset.*

3. Quality is defined as dataset provenance which relates to dataset authoritativeness and trustworthiness, accuracy which refers to data correctness, completeness which refers to "no missing values", cleanliness which refers to "syntactic correctness", methodology which refers to the ways data was collected, and timeliness which refers to the "update frequency (freshness)" of data. The percentage of ranking these dimensions by the participants vary.

- *Provenance:* It refers to data authoritativeness and trustworthiness of the data publisher. *In our model, we assume that a contextual contract unit registers information of data provider and data requester whenever dataset is accessed. In this thesis, we provide the model and elements of the contextual contract unit, but the implementation of rating provider is out of the thesis scope.*

- *Accuracy:* It is defined as data correctness. *In our opinion accuracy can be defined as either data truthfulness (data content) or the metadata correctness in terms of the level of matching of the metadata description to the actual dataset content. Veracity feature reflects truthfulness, and is part of our dataset context model. Our method requires data providers to publish metadata and we assume that their descriptions fully reflect the dataset content.*

- *Completeness:* It is related to percentage of missing data values in the dataset. *In this thesis completeness is related to sufficiency of data in the dataset to meet the stated goals of the analyst.*

- *Cleanliness:* It refers to level of format errors and spelling mistakes included in the dataset. This feature is content based and if datasets are managed in another data source these features should be managed and cleared. Automatic data pooling might lead to these errors more. *We assume that dataset providers would be able to manage this task.*

- *Methodology:* The paper states that this feature is not considered mandatory by most of the participants. *In this thesis we require the dataset provider to include information on "from where and when" the data was collected, and do not require information on the methodology used for collecting it.*

- Timeliness: This is an important dimension to be included in metadata description. *We believe that the inclusion of Velocity as a function of "Update Frequency" in our proposed dataset context model will characterize timeliness of datasets.*

Table 2.1 summarizes the list of elements recommended in (Koesten et al., 2020) for quality attribute descriptions are given. We will make use of these elements for our formal metadata model description.

2.2.2 Experience Report-Clinical Research (Bian et al., 2020)

As opposed to the in-depth experiment on dataset quality attributes done in (Koesten et al., 2020), this paper reviews many published research papers on data quality (DQ) dimensions and assessment methods in "a national clinical research network". Because

Table 2.1: Elements needed by Data seeker to Understand Datasets (Koesten et al. (2020))

	Relevance	Usability	Quality	Summary Description
scope	Required			
Granularity	Required			
Context	Required			
Documentation	Required	Required		
Format		Required		
Comparability		Required		
References		Required		
Access		Required		
Size		Required		
Language		Required		
Provenance			Required	
Accuracy			Required	
Completeness			Required	
Cleanliness			Required	
Methodology			Required	
Timeliness			Required	
Title				Required
column names				Required
Key attributes				Required
attribute datatype				Required
statistics				Required
Geographical area				Required
Geographical area				Required

Electronic Health Record (EHR) datasets are now part of Big Data, and their quality affects the outcome of patient analysis we decided to study the survey (Bian et al., 2020) and observe the commonalities as well as the specific concerns on what contributes to DQ according to different domain experts.

The paper singles out the three widely cited papers (Chan, Fowls, & Weiner, 2010; Khan, Callahan, Barnard, & et al., 2016; Weiskopf, Hripcsak, Swaminathan, & Weng, 2013) in DQ literature as their primary sources, although they have indeed covered much broader spectrum of papers published on DQ during the last 25 years. Below we distil the essence of the DQ dimensions, DQ assessment methods, and the mapping of data characterization checks on the these two DQ aspects.

The paper has selected 14 DQ dimensions from their literature survey and provided their definitions. The dimensions *Plausibility* has 3 sub-dimensions (*uniqueness*, *Atemporal*, and *Temporal*), and *Conformance* has 3 sub-dimensions (*Value*, *Relational*, and *Computational*). The rest of the dimensions are atomic, From this list we notice the following:

- Majority of researchers have identified the DQ dimensions *Currency*, *Correctness/Accuracy*, *Plausibility*, *Completeness*, and *Value Conformance* as most important.

- Least number of researchers have identified *Compatibility*, *Security*, *Information Loss and Degradation*, *Consistency*, and *Understandability* as necessary requirements.

- The DQ dimensions *Relational Conformance*, *Computational Conformance*, and *Flexibility* have been identified by a few researchers as significant for EHR structure (relational) and shareability.

The paper has identified 10 DQ assessment methods. Majority of research papers review in (Bian et al., 2020) have identified the method to check *Element Presence* as the most important method. This method is to determine *whether or not desired or expected data elements are present in the dataset.* This method is related to the DQ dimensions *Completeness* and *Conformance.* The next highly ranked test is *Conformance Check*, which is related to the dimensions *Uniqueness Plausibility*, *Completeness*, and *Value Conformance.* The DQ assessment method *Validity Check* is related to the DQ dimensions *Conformance* and *Atemporal plausibility.* The methods *Log Review*, *Data Source Agreement*, and *Distribution Comparison*, and *Qualitative Assessment* are specific to statistical databases, and are often adapted for EHR management. The method *Gold Standard* refers to "data extracted from paper records, such as questionnaires and hand written notes of doctors", and is specific to healthcare domain. The paper identifies only one researcher mentioning the assessment methods *Qualitative Assessment* and *Security Analysis.* Based on these reviews, the authors (Bian et al., 2020) have the following key conclusions.

(1) The definitions of and the relationships among the identified DQ dimensions are *not clear*. The authors quote a few other works to suggest that *accuracy* and *correctness* dimensional are not the same.

(2) The set of DQ dimensions are not *independent.* The authors state that *comparability* dimension overlaps with *Completeness*, and *Consistency.*

(3) The practice of DQ assessment is *not evenly established* among the practitioners. This observation is justified because most used data checks are *element check*, *validity check*, and *conformance check* and these target the DQ dimensions *completeness*, *conformance*, and *plausibility.* Rest of the DQ assessment methods and DQ dimensions are very rarely used.

(4) The authors argue that future work is necessary on understandable, executable, and reusable DQ measures and their measurements.

2.3 Selection of Data Sources - First Set of Naive Approaches: Publications during 2000-2015

The first work to recognize the "quality" aspect for data source selection was published in 2000 (Mihaila et al., 2000). In this work it was recognized that access to data was hindered by many challenges related to locating data sources and by lack of facilities to determine the quality of those that were located. They proposed to solve this problem using *Quality of Data* (QoD) parameters such as completeness, recency, frequency of updates, and granularity. They introduced a metadata model for source content and quality aspects, defined a query language for users to query this model and select data sources. This approach focused more on query structure (similar to SQL) while XML was their choice for meta model. The paper gives an informal analysis of their solution, explains the *exponential* growth in (1) the process of identifying data sources, (2) finding the right combinations for those data sources, and (3) determining the relevance. This approach was an eye opener to other important aspects in BD analytics. However, due to the exponential complexity, it was not followed up by any subsequent researcher. Due to the absence of any followup work on this approach by the authors, we assume that this approach was not formalized, perhaps not fully implemented, and/or validated.

A decade following the publication of the above seminal work, four papers on "Selecting Data sources" have been proposed during 2012-2015. In 2012, the authors (Dong et al., 2012) approached the data source selection problem considering the motto "Less is More". That is, they focused on reducing the number of the selected data sources in hopes of boosting up performance (reduce time complexity) of the process. Towards this goal, they used a probabilistic model to determine the utility of "adding a new data source" to a set of already collected data sources. The theory behind this approach is based on *Marginalism Principle* of Economic Theory. They used certain probability distributions from the Economic Theory of Marginalism to estimate "source data accuracy" for selection and addition to their collection, assuming the "independence of data sources, and that the data items are *not distinguishable* in terms of error rate and false value distribution". They gave simulation results of the theoretical approach, and commented as follows on the accuracy and complexity of this approach.

- The accuracy results depend on their *knowledge of source accuracy and distribution of false values.* So, there is no firm theory to support it.
- The hardness of accuracy computation is an *open problem.* They claim (no proof is given) that it is #P-hard. However, they also state that *if a polynomial time oracle is used for fusion accuracy*, some of their algorithms have Polynomial time.
- They state that *solving Marginalism problem is NP-Complete.*

From these statements, we conclude that the twin problems reported in (Mihaila et al., 2000) on the exponential time performance and the explosive growth in the number of data sources to satisfy sufficiency of user requirement were not solved in the Marginalism-based probabilistic approach (Dong et al., 2012).

The subsequent work of this group (Rekatsinas, Dong, & Srivastava, 2014) is more of a continuation to their previous endeavour with similar approaches. One of the papers addressed "dynamic data source selection" problem with time-dependent metrics for "freshness, accuracy, and coverage", however no specific result was reported. In (Kornegay & Segal, 2013), the authors emphasized on "comparative effectiveness research" and that the choice of data must be "driven by the research question (goal)". They provided a guidance and a set of key considerations for data source selection. This paper is more of "a policy type" for data source selection without providing a specific method to solve data source selection problem. The paper (Rekatsinas, Dong, Getoor, & Srivastava, 2015), authored by two of the authors of the previous papers (Dong et al., 2012; Rekatsinas et al., 2014), suggested that *the data source management system must automatically assess data quality*, without offering any specific method/suggestion to achieve it. The proposals provided in the above paper referred four other papers that provided valuable discussions and theories in data source selection topic. However, no deterministic practical approach to solve data source selection problem was proposed.

2.4 Selection of Domain-specific Data Sources - Constructive Approaches Published during 2017-2018

The papers discussed in this section report the rationale and conceptualizations that lead to effective implementations for domain-specific data selection. The work in (Sansone et al., 2017) reports the research conducted by an international group of researchers, governmental organizations, service providers, and National Institute of Health (NIH) in the discovery of available *biomedical* datasets. The work in (Strozyna et al., 2018) describes the selection, and retrieval of datasets in *maritime* domain using Delphi method Grime and Wright (2016); Hsu and Sandford (2007); Okoli and Pawlowski (2004). The Delphi method is based on a set of procedures manually done by experts in maritime domain to select quality features for assessing dataset quality. Below we summarize these two significant breakthroughs,

2.4.1 Dataset Selection in Biomedical Domain

The research approach in (Sansone et al., 2017) uses the NIH catalog of biomedical data which was compiled and published in 2015 in the Data Discovery Index (DDI) (Bourne, 2015). Its prototype, called DataMed, was launched in 2016 through which many currently available biomedical data sources are browsed and searched. The paper explains the design and implementation of Data Tag Suits (DATS) model that supports DataMed data discovery index. The DATS and DataMed are related and complementary, in the sense that the DATS meta model elements are used for indexing and searching in DataMed. The DATS core elements are generic and are applicable to any type of dataset. Some of the elements are specific for life, environmental and biomedical science domains. The paper states the FAIR principle of *Findability* (F), *Accessibility* (A), *Interoperability* (I), and *Reusability* (R) as their core objective in designing DataMed, which uses DATS. The paper gives three considerations in DATS model development. The first is the search and review of many metadata models and schemas. The second is the analysis of many use cases (top down manner) and mapping/integrating schemas (bottom up) to reach a convergence on common metadata elements. The third is using the collected sets in the previous two steps. Thus, the entire work was guided from a software engineering-based pragmatic goal of putting together a metadata model consistent with FAIR principle to enable researchers in biomedical sciences access available dataset repositories. No new dataset selection engine was developed by the team. Instead, DATS was made available as an "annotated serialization scheme in schema.org", which in turn is used by several search engines like Google, Yahoo, and Microsoft. The notion of "relevance" (or quality) was not central to their effort.

2.4.2 Dataset Selection in Maritime Domain

The method proposed in (Strozyna et al., 2018) is the first one to develop a "fully working dataset discovery method" based on quality criteria. The paper demonstrates with an example how datasets of relevance from open datasets in maritime domain can be retrieved to best match the given quality criteria. Their method uses Delphi method (Grime & Wright, 2016; Hsu & Sandford, 2007; Okoli & Pawlowski, 2004) that involves experts to decide quality criteria, and assign weights to quality attributes consistent with their levels of significance, and manually select datasets based on a numerical measure of significance computed from the assigned weights. As such "no engine" was developed. However this work makes a significant contribution to designing dataset discovery engines based on the following principles:

- *User-centric Quality-based Access:* Each potential data source should be assessed taking into account user's requirements and a set of selection criteria. This principle we call *Contextual Compatibility*, and is the basis for the design of dataset search engine.

- *Assuring Safety and Security:* Maritime domain must be secure and safety-sensitive in its operations. So, the paper identifies the domain-specific safety and security requirements, and postulates that open data sources must be scrutinized by experts and select datasets that do not violate safety and security requirements. We follow this principle in the thesis, make data source providers include them as part of source context information to alert the authentication levels required of the users for retrieving datasets.

- *Adherence to Standards:* The authors acknowledge that in maritime domain "there was no standards or procedures to dictate the set of quality criteria". So, they brought in "domain experts" and followed strictly the steps in Delphi Method. We follow this principle, requiring data source providers include standards followed in metadata part and legal requirements as part of data source context.

- *Choice of Quality Attributes:* Following Delphi procedure, the experts choose the quality attributes *Accessibility*, *Relevance*, *Accuracy & Reliability*, *Clarity*, *Timeliness & Punctuality*, and *Coherence & Compatibility*. They associated a precise meaning (semantics) to each quality attribute, and assigned "weights" to the quality attributes that denote their "level of importance" in the selection. These are shown Table 2.2. We follow this principle, and let the user (analysis team) specify the set of quality attributes desired, and choose attribute weights with other user-level semantics to increase precision of selection. Thus, user-centric contextual interaction in dataset selection process is strengthened.

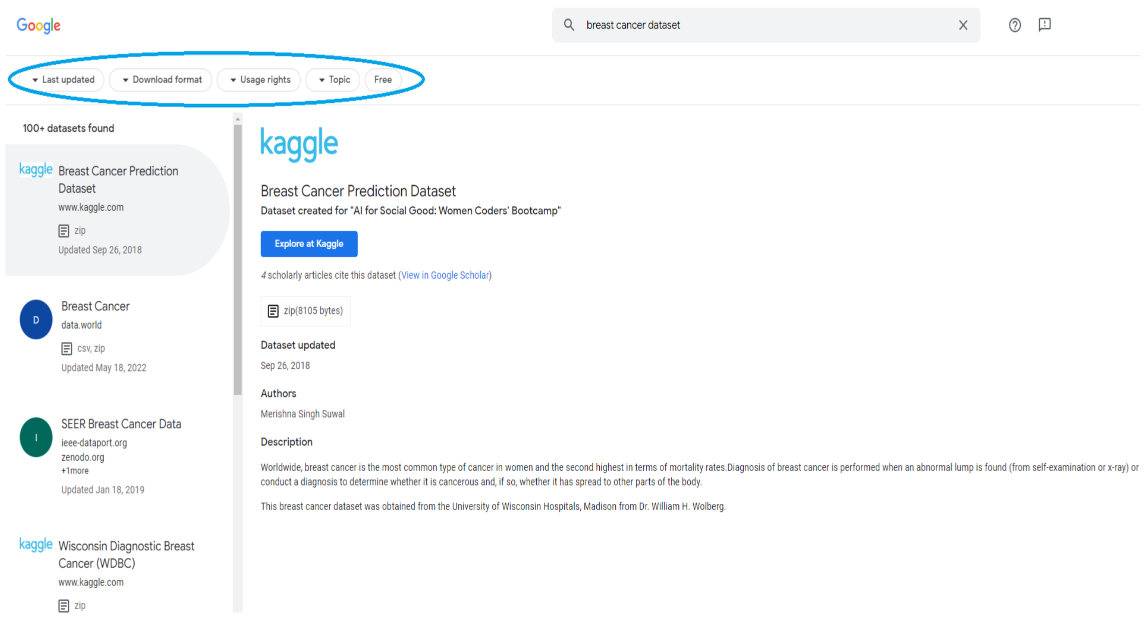
A brief summary of the rest of the Delphi procedure is as follows: The Delphi approach was used in two rounds to assess each data source. In the first round, the experts received data source information and some statistics. Based on it, each expert assigned “marks (grades)” to quality attributes in each data source. The marks and their numerical equivalents are as follows:

$$\text{High} = 3; \quad \text{Medium} = 2; \quad \text{Low} = 2; \quad \text{N/A} = 0;$$

A weighted mean of the assigned scores for each data source was then calculated and the data sources were ranked in decreasing order of the scores. It is clear that the Delphi method was a “manual collaborative work of experts” on the data sources (their metadata and statistics). The Delphi approach has several merits when applied to a very specific domain in which the number of open data sources is small and experts are part of analysis team. The drawback of this manual approach is that both cost and time will increase when the number of datasets increase, or when the set of experts change. In this thesis, the framework component at the user (analyst) end can deal effectively with the later type of change by triggering a repetition until a consensus is reached to submit the user requirements to the automated dataset search engine. The framework component at dataset provider end can deal with updates/changes at the data source level, independent from the user level framework component.

Table 2.2: Quality Measures - Delphi Method (Strozyna et al., 2018)

Attribute Name	Informal Semantics	Weight
Accessibility	Possibility to retrieve from website	0.3
Relevance	How well the data can suit the requirements	0.3
Accuracy & Reliability	Scope, Coverage and Completeness	0.2
Clarity	Precise Description of Data model and Data Provider	0.1
Timeliness & Punctuality	Update Frequency and Publishing Delay	0.05
Coherence and Compatibility	Definition of Data and Unit of Measurement	0.05



100+ datasets found

kaggle Breast Cancer Prediction Dataset
www.kaggle.com
zip
Updated Sep 26, 2018

Breast Cancer
data.world
csv, zip
Updated May 18, 2022

SEER Breast Cancer Data
ieee-dataport.org
zenodo.org
+1more
Updated Jan 18, 2019

kaggle Wisconsin Diagnostic Breast Cancer (WDBC)
www.kaggle.com
zip

kaggle

Breast Cancer Prediction Dataset

Dataset created for "AI for Social Good: Women Coders' Bootcamp"

4 scholarly articles cite this dataset (View in Google Scholar)

zip(8105 bytes)

Dataset updated

Sep 26, 2018

Authors

Merishna Singh Suwal

Description

Worldwide, breast cancer is the most common type of cancer in women and the second highest in terms of mortality rates.Diagnosis of breast cancer is performed when an abnormal lump is found (from self-examination or x-ray) or conduct a diagnosis to determine whether it is cancerous and, if so, whether it has spread to other parts of the body.

This breast cancer dataset was obtained from the University of Wisconsin Hospitals, Madison from Dr. William H. Wolberg.

Figure 2.1: Google Dataset Search: Search Result

2.5 Dataset Search Engines: Recent Papers Published During 2019-2022

We review in this section the most recently published papers (at the time of writing the thesis) that discuss "Dataset Search Engines" (DSE). We discuss them in the chronological order of publication.

2.5.1 Google Dataset Search Engine

In Noy et al. (2019) the authors explain a DSE developed by Google. A user who wants to access Datasets has to access the web page `https://datasetsearch.research.google.com`. As in general Google web page access, accessing the above link opens a new window in which the user enters a set of keywords, and initiate the search. In response, the Google DSE window, shown in Figure 2.1, opens. It provides many filters such as "Last Updated", "Usage Rights" etc. When the user selects one of the filters, the Google DSE displays a list of the metadata that are relevant to the selected category. Each metadata in the list has a link through which the user can access the dataset associated with that metadata.

The internal details on how these metadata are fetched by Google DSE are as follows. The regular Google search engine uses its known crawlers Najork (2009) to traverse through the web in order to fetch the metadata of every dataset that it can access. Then, these set of metadata are cleaned and normalized, reconciled, indexed, and stored in a Google DSE repository. The repository is updated frequently (periodically) by Google to add metadata of any newly identified datasets on the web. When the user selects a "filter" after entering the keywords on the web page `https://datasetsearch.research.google.com`, all metadata in the Google DSE repository that match the keywords are selected, and the set of selected metadata is filtered to extract only those metadata set that match the chosen filter category. This filtered set of metadata is displayed in some order to the user, who can follow the link in each metadata in this set to access the dataset associated with that metadata from its original repository. Thus, the Google DSE provides an "open ecosystem" that connects users to the datasets that exist in their repositories.

The two important deficiencies in this process are *contextual incompatibility* and *incompleteness*. The former arises because the context of getting metadata description of a dataset is different from the context of user getting access to the dataset content. The later arises because request through keywords could have been made before the updates of Google DSE repository and the dataset repository, which are necessarily asynchronous, not every dataset will necessary include all the data relevant to user request. The paper is "silent on this issue". It does not mention nor give any hint about cleaning outdated datasets from its repositories. The paper states that "Metadata for datasets is generated automatically from RDF, JSON, and W3C files". As pointed out in (Castelo et al., 2021) "published metadata may be incomplete and in some cases not compatible with the actual description of metadata in a data source". Although Google claims that it excluded from its repository any metadata that does not match dataset profile page, it does not seem to solve the incompleteness issue.

The paper states that in order to get an accurate and complete dataset metadata description, data owners should be motivated "to find it valuable and rewarding not only to publish their data but also to describe its metadata better and more fully". However, data owners are not provided with a formal standard template by Google that could guide them to build and publish dataset metadata in a standard formal format. Instead, data owners have the choice to select some or all metadata elements defined in one of the available standards such as `schema.org`, and DAT to build their metadata. This leads to inconsistent metadata descriptions of data owners and in turn increases the level of difficulty of users in accessing datasets, and understanding dataset quality and relevance. In addition, uniform formal approach for matching and ranking of datasets is not possible with heterogeneous and inconsistent metadata descriptions.

An important observation of the web page `https://datasetsearch.research.google.com` is that Google DSE does not provide a filter for "dataset quality", an important requirement of DSE (Strozyna et al., 2018). So, users cannot input quality aspects for attributes, weights, and keywords to express their level of significance. Moreover, "relevance factor" is not made precise in Google DSE, and ranking in Google DSE is not user-centric because the regular Google engine uses its own criteria for ranking.

2.5.2 Auctus - Dataset Search Engine

Auctus (Castelo et al., 2021) is an "open-source" DSE that was designed to support *data discovery, augmentations*, and a rich set of "discovery queries" that combine multiple constraints on keywords, data types, temporal and spatial data. The functionality of the user interface `https://auctus.vida-nyu.org/` is rich enough to allow users upload their own datasets, use keywords and different constraints such as date range and geographical area to compose queries to search for new datasets. Auctus provides operations for data augmentation and data integration. Hence, users can perform join or concatenate operations on the datasets they input to Auctus with datasets discovered by Auctus DSE through querying. From these stated goals, it is clear that Auctus DSE seems more complex and more powerful than Google DSE.

The paper gives an overall summary of Auctus DSE architecture, explains the functionality of User Interface, and describes two use cases. Specific details on architectural components, standards used for metadata organization, algorithms of search methods used, and how the secondary data for use cases are fetched for analysis are not given in the paper. So, our survey below is limited by the lack of details in the paper.

The architectural components are *Data Ingestion*, *Data Profiling*, *Storing Data and Data Summaries*, *Indexing*, *Querying and Ranking*, and *Augmentation*. Auctus uses APIs to retrieve datasets from their repositories. Currently, it ingests data from user input datasets, and a few open data source platforms such as *https://www.worldbank.org* (World Bank Open

Data) , and *https://socrata.com*, the platform for open government data. Once data is ingested, profiles are constructed. This includes the structured data with typed attributes and type-dependent statistics. Next, Auctus automatically builds "data summaries" of datasets input by the user and those fetched from external repositories. The summaries (metadata), instead of the full datasets, are stored. The paper does not reveal the metadata structure constructed in Auctus to store the summaries, although the authors claim that these are "indexed in an Elasaticsearch" server and used during the search phase. Auctus does not prefer using the metadata of datasets available on the web, as it limits the functionality of the DSE. It's view is that the metadata generated automatically from the web is not efficient and will mostly result in limited, incomplete, and inaccurate information of datasets. In fact, the paper states clearly that "automatically generated metadata can be incompatible with the actual dataset". Therefore, Auctus uses its own automatic metadata generator to construct its dataset content summaries and other related statistical information. This seems to be the main reason why Auctus is limiting itself to a small number of datasets in repositories as *data.alaska.gov*, *https://www.worldbank.org*, and *https://socrata.com*. Having said that, Auctus gives the dataset providers the freedom to "correct mistakes found in the metadata generated by it on the fetched datasets". This implies that the authors believe that the Auctus automatic metadata generator is neither complete nor correct.

The paper does not give the query structure in Auctus, but from their account of query processing we guess that Auctus querying structure resembles that in a relational database system. "Join Search", and "Union Search" are the two query types that the DSE uses in dataset integration and augmentation. Join search finds other datasets from the repositories of its search platforms that can be joined with the user input datasets. Union search finds datasets from the repositories of its search platforms that can be concatenated with user input datasets. No detailed description of algorithms behind these search methods are given. The paper states that (1) Join search results are ranked based on the "intersection of the summaries", and (2) Union search results are ranked based on the *Levenshtein similarity between the names of the matching attributes* (S. Zhang, Hu, & Bian, 2017). The paper does not give details of the underlying search techniques. As such we are unable to evaluate them and comment on their merits. The best guess we have can be summarized as follows:

- The open source platforms that Auctus has selected to focus for their study contain datasets that are exclusively “textual information-based content” which can be typed using “nominal, categorical, and string” types.. Auctus has not selected any open dataset repositories from scientific (healthcare, biomedical) and/or critical domains such as maritime domain wherein ontology will be required to define semantics of concept terms. So, the specific operations that are included in the DSE are “borrowed” from information processing domain.

- For ranking results that use “Join search”, Auctus DSE uses “intersection size”, and for ranking results that use “Union search” Auctus uses the Levenshtein distance measure. The paper does not explain how “Intersection size” on metadata of datasets in “Join operation” is calculated. Levenshtein distance measure is a string metric for measuring difference between “two sequences of characters” (two words). Informally, the Levenshtein distance between two words is the *minimum* number of single-character edits (i.e. insertions, deletions or substitutions) required to change one word into the other. They use it on “attribute names” (in the datasets). This measure is popular/acceptable in information processing when applied to “keywords” of documents. This measure is not suitable for comparing and assessing distance between concept terms (strings) in biomedical or healthcare or safety-critical domain.
- From these two remarks, it follows that both ranking criteria are not suitable for all application domains. In particular, these criteria are based only on “structural” (textual) details of metadata and not on semantics of metadata and as such they cannot be used in ontology supported safety-critical domains such as healthcare, maritime transport, and biomedicine.

Another important observation is that the paper does not discuss “quality attributes” and the challenges related to discovering datasets that are relevant to “user requirements”.

2.6 Attribute and Schema Based Dataset Selection - Recent Papers Published During 2020-2021

Whereas a DSE can be used by different types of users (ML groups, data practitioners, students) to find available datasets that are open source, the two methods discussed in this section describes methods that are specific to attribute matching or schema matching targeting pools of datasets in "tabular forms" (column-wise described by attributes). As such, these methods can be used within a DSE.

2.6.1 Dataset Discovery in Data Lakes - Information Retrieval-based Approach

The paper (Bogatu et al., 2020) proposes five types of evidences to match an attribute specified by a user to an attribute of a dataset available in a *Data Lake*. The papers suggests, without giving a definition/chracterization, "what a data lake is". The recent book (Baum, 2020) characterizes a data lake as a "centralized repository constructed by an enterprise to host its raw, unprocessed *enterprise data* for its specific organization analysis goals. That is, a data lake is owned by an enterprise in which unedited and un-summarized data from all operational sources of the enterprise will be deposited. Some data may be structured. For each dataset put in the lake, its schema, metadata and content will be included. This book explains in detail the many challenges in managing, sharing, and retrieving data from a data lake, because different domain experts from the enterprise who own the data lake "look into the datasets only after they are thrown into it". What we are focusing in this thesis is the design and development of a DSE that can be used by a variety of users to find and select "relevant" datasets available in open datasets with or without registration. Hence, the scope of research in data lakes do not suit the scope of our thesis. So, our interest in this paper is limited to examining whether their attribute-based retrieval of datasets is of any relevance to our goal.

A user query in their methods is a set of attributes targeting datasets whose attributes might match the query attributes. The five evidences (query types) for attribute based retrieval are (1) attribute name, (2) attribute values, (3) word–embedding for semantics, (4) format, and (5) domain distribution similarity. They use Jaccard function, which defines the similarity between two *feature sets* X and Y as $\frac{|X \cap Y|}{|X \cup Y|}$, to measure matching for methods (1), (2), (4), and (5). The methods used in the paper involve hashing functions, extracting k-grams, and other approximations and then use Jaccard function and "Cosine Similarity Function" to calculate a measure of similarity between user specified attribute set and the attribute set of a dataset in the lake. Jaccard measure is purely a "structural set-theoretic" measure, with no semantics involved. It is not a distance-based (which requires geometric modeling of data), although it is used that way in the paper. To measure the similarity of matching for method (3), they use "Cosine similarity function", which is defined for two *numerical vectors* $A =< a_1, a_2, \cdots a_n >$ and $B =< b_1, b_2, \cdots b_n >$ as

$$\frac{a_1.b_1 + a_2.b_2 + \cdots a_n.b_n}{\sqrt{a_1^2 + a_2^2 + \cdots a_n^2}.\sqrt{b_1^2 + b_2^2 + \cdots b_n^2}}$$

The paper mentions that they conducted 10 experiments with different case scenarios of attribute distribution and calculated *precision* and *recall.* They claim that using this method the accuracy of precision and recall increased about 25% compared to previous methods. We observe the following issues on their methods, that make their methods not suitable for our work.

(1) Their methods are based on content information, not on metadata information.

(2) Attribute matching uses information retrieval methods, such as extracting "tokens" (in this case k-grams)before applying Jaccard measures on the sets of tokens. There is no justification to choose a specific value of k. and hence the methods are only approximate. The authors admit that "*we adopt an approximate solution that offers efficient distance computation at the expense of accuracy.* Loss of accuracy will lead to erroneous conclusions, which in many critical domains such as healthcare, maritime transportation, and finance are not acceptable.

(3) Semantics of attributes is not taken into account. Purely set theoretical calculations (set intersection) on attributes that are "synonyms" or related through an ontology (disease types or drug types) will produce unacceptable results in healthcare domain.

(4) The authors use "set model" for attributes for methods (1), (2), (4), and (5). So, neither the order of attribute specification nor the number of attributes specified in a query do not matter. However, for method (3) they use "Cosine similarity function", which require "both vectors have the same order of attributes information" and "the length of the vectors are equal". This requires that the numerical measures they calculate from user query attributes are consistent with the order and in size as the

Table 2.3: Schema Matching Methods Used in Valentine Method

Method	Description
Cupid : schema based (J, Bernstein, & Rahm, 2001)	Linguistic and structure level
Similarity Flooding:schema based (Melnyk, Molina, & Rahm, 2002)	Graph-based string matching
COMA (Do & Rahm, 2002): schema & instance based	Graph based & human feedback
Distribution: instance based (M. Zhang, Hadjieleftheriou, & et al, 2011)	Value Distribution using Mover's Distance (EMD)
SemProp: external knowledge (Fernandez & et al, 2018)	Capture relationship between schema elements Using Domain specific Ontology
EmbDI: external knowledge (Cappuzzo, Papotti, & Thirumuruganathan, 2020)	Based on the relationship between columns of two datasets comparing corresponding embedding

values they calculate from attributes of datasets. However, their discussions do not bring out this consistency.

2.6.2 Valentine: Dataset Selection Based on Schema Matching Method

Valentine (Koutras et al., 2021) provides dataset discovery method based on schema matching methods. The paper proposes a dataset discovery method to use collected datasets for data augmentation. That is, Valentine takes a dataset as an input (Target) and through their matching techniques find the matching datasets. These matching datasets could be integrated with the Target dataset. The matching method of Valentine uses six existing matching techniques for schema matching. The six methods are categorized into two categories. The first category is "schema-based", which means it needs information provided on the schema level to perform the matching process. The second category is "instance-based", which needs information content of the schema level such as column values and other external knowledge. In Table 2.3 the six methods used in Valentine are briefly explained. A summary of the conclusion of the authors (as is from the paper) is the following:

- *One size does not fit all.* "Our evaluation over both Valentine's fabricated dataset pairs and those stemming from real-world data show that there is not a single schema matching method that consistently performs better than others. Instead, we see that COMA exhibits higher effectiveness over most of our fabricated dataset pairs, yet the distribution-based method is the most well-suited for our real-world ING datasets. Consequently, we believe that COMA's approach of composing state-of-the-art matching methods (e.g., by adding the recent embedding-based approaches), should be the preferred way in dataset discovery or other integration pipelines."

- *Embedding for matching.* "Our experimental results showed that SemProp's pre-trained embedding provide low effectiveness when used in isolation. On the other hand, EmbDI's local embedding can improve effectiveness, yet most of the times they do not perform as well as other state-of-the-art schema or instance-based methods. Therefore, while we acknowledge that embedding-based techniques can improve effectiveness by incorporating them into existing matching methods, we believe that further research is needed in order to make them effective."

- *Complex parameterizations.* "Most methods require complex parameterizations in order to perform well. For most part, parameters are dependent on the input data that needs to be matched, which makes it very hard for practitioners to use those methods. We believe that our community should focus on "self-driving" matching methods that do not require parameterizations. Machine learning might be a solution to some of the parameterizations problems, but they would require at least some availability of ground truth to steer the learning process."

- *Simple baselines perform well.*" Our simple baseline Jaccard-Levenshtein matcher (S. Zhang et al., 2017) (70 lines of Python code) works surprisingly well, especially considering its simplicity. We argue that similar baselines to ours, along with the rest of the methods discussed in this paper, can foster future comparative analysis for schema matching and dataset discovery processes."

- *Humans-in-the-loop.* "Self-driving matching methods should be able to work alongside humans giving feedback on the matching process, not in the form of parameters or thresholds, but in the form of positive/negative examples. In the same spirit, the design of schema matching methods should focus on presenting matches as ranked candidates. We strongly believe that the schema matching problem should be approached as a search problem, rather than an optimization problem. Schema matching of the future should focus more on preparing results that will be shown to humans, and should utilize feedback from humans.

- *Schema Matching is resource-expensive.* "Instance-based methods are still expensive as they have to calculate similarity metrics between large sets where it can be very time-consuming to find matches. Future research should focus on approximate methods to allow for better scaling."

Based on the review of Valentine method and the methods enumerated in Table 2.3, we have two critical observations on "good attribute-based" and "content-based" data understanding for dataset selection. In our opinion, data understanding is not enhanced or made easier by any of the methods enumerated Table 2.3.

- *Attributes:* The paper states that data understanding is improved by "good attribute naming" for schema-based methods. However, in our opinion it is more related to "understanding the semantics and context" of the datasets. To elaborate, let us take the terms "prescription" and "medication", where "prescription" is used in the dataset $DS1$ and "medication" is used in the $DS2$. Both terms have been used interchangeably in many datasets, and hence may be regarded as "good attribute names" according to the authors of this paper. However, if we use Wordnet Ontology used by Valentine researchers (Koutras et al., 2021) the similarity measure between the terms "prescription" and "medication" is 0.3, which is rather low. That is, they are not "good equivalent terms" according to Wordnet. So, when these terms are used in different datasets, we cannot assume that they provide the same meaning (semantics). This semantic issue cannot be resolved through typing. If they are assigned different types, their "dissimilarity" will increase. It will not get better if they are assigned same type information, because assigning same type might increase similarity measure, but not to the level when they have same structure and semantics.

- *Content:* As observed earlier, the authors admit that "instance-based methods are expensive". In our review (Batini et al., 2009; Chapman et al., 2020; Dong et al., 2012;

Ottenheijm, 2017; Paleyes et al., 2022), we have already brought out the complexity and incompleteness on content-based analysis. Moreover in several countries, due to privacy and social aspects, dataset content may not be accessible and hence data enhancement methods are irrelevant.

Chapter 3

Design Methodology of Tightly Coupled Framework (TCF)

In Chapter 1 we motivated the need for TCF in the discovery of datasets that match the goals of an analysis team. In this chapter, we give an overview of the methodology for a component-based design of TCF. First we motivate as to why a component-based methodology is suitable for TCF. Next we give an overview of the components and their interfaces through which they provide and request services in achieving dataset discovery that meet the requirements of analysts. The detailed designs of components and their interfaces are discussed in later chapters.

3.1 Justification for Component Design

From the survey done in Chapter 1, two design issues arise. The first design issue is modeling the three entities *Dataset Provider* (DSP), *Analysis Team* (ANT), and *Data Search Engine* (DSE) that arise in a dataset discovery problem. The role of entity DSE is to match the analyst requirements (from entity ANT) with datasets provided by the DSP entity, retrieve and rank the datasets. The second design issue is the necessity of a "quality model" associating the quality dimensions and quality assessment methods with the entities who will have the mandate to discharge them. A component design is a natural choice to fulfill these two design issues, as justified below.

- *Encapsulating Role and Responsibilities:* Each entity has a role and a set of responsibilities, which can be fulfilled in a "service-oriented" paradigm. So, each entity is encapsulated into one component. Each component provides the set of services defined by its role, and requests services from other components to discharge its obligations. Through interfaces, whose specifications define the set of services requested (received), the components interact.

- *Enforcing Safety and Security:* At interfaces, security and safety constraints can be enforced.

- *Reliability and Availability:* Through interfaces, a component can alert its neighbours (those connected to its interfaces) on the "service failure and delay" of services at that interface.

- *Attributes, Data Types, and Constraints:* These elements can be defined locally within each component. An attribute qualifies the semantic content associated with an element defined within a component. A data parameter is information carried through a service (requested or received). It includes name, data type, and value. A constraint is a logical expression, defined over data parameters, and locally defined attribute values.

- *Quality Model:* In the survey Chapter 2 we have identified a broad set of dataset quality dimensions, but none of the surveyed papers have put forth a model of associating the set of quality dimensions with datasets. In this thesis, a component can select a subset of quality dimensions from a predefined set of quality dimensions in order to meet its goals.

Figure 3.1 shows the component design of TCF framework. Corresponding to the three entities DSP, ANT, and DSE we respectively design the components *Provider Component*(PrC), *Analyst Component*(AnC), and *Coupling Engine Component*(CeC). These components respectively encapsulate the tight coupling of the three entities. From the explanation given above for choosing component design, it follows that the term "tightly coupled" refers only to the "contextual binding" that the interfaces of component CeC provide between the components PrC and AnC. We introduce the *Ontology Component*(OnC) to provide "expert

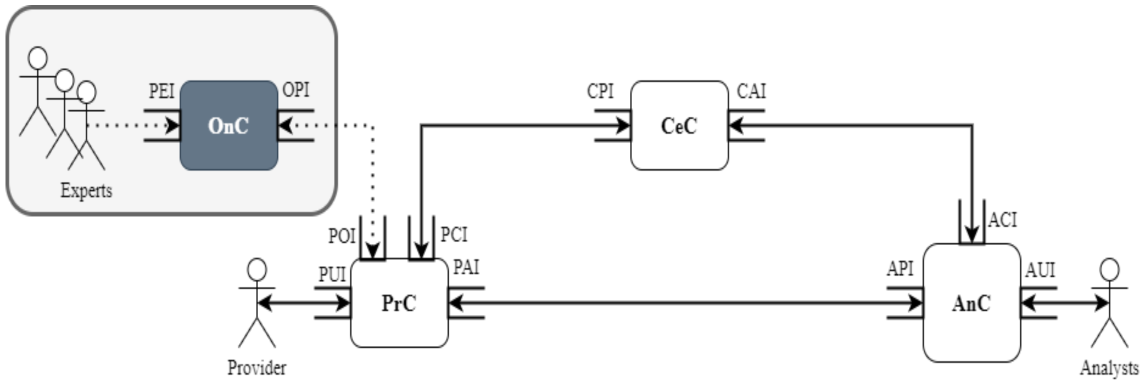


Figure 3.1: TCF Framework Component-based Design

support" to PrC, as briefly explained in Section 3.2 and further elaborated in Chapter 4. From the software design point of view, the components are "loosely coupled", because the internal structural aspects of each component and its behavior-preserving refinements are done without affecting the behaviour of other components.

3.2 An Overview of OnC Component

OnC is a black box of the TCF framework. We can informally regard this component as a big data storage in which the set of domains, set of sub-domains for each domain, and ontologies created by experts for each (domain,subdomain) pair are stored. We assume that this activity has been done by experts through OEI interface of OnC, and is available for TCF. The research in ontology creation for any application domain is a vast study in itself. It is beyond the scope of the thesis to go into implementation of OnC. So, following the works of many authors (Chandrasekaran & Mago, 2021; Trojahn, Vieira, Schmidt, Pease, & Guizzardi, 2021) who use a specific ontology for a *specific* application within a specific *domain*, in Chapter 7 we choose an ontology (Seng et al., 2021) for Type 2 Diabetes, implement it, and illustrate how it is used to support metadata descriptions. For the sake of clarity on the "expert help" that OnC provides to PrC, we will explain in Chapter 4 the functionality of OnC and how PrC interacts with it through the interface OPI.

3.3 An Overview of PrC Component

Although there are different methods to automatically build metadata for datasets available on the web using standards such as Schema.org and DCAT (Koutras et al., 2021; Noy et al., 2019), all experience reports (Ashmore et al., 2015; Koesten et al., 2020; Noy et al., 2019) have said that these generated metadata descriptions need a big improvement. As we pointed out in Chapter 2, Auctus (Castelo et al., 2021) allows users to upload their datasets, and it generates the metadata information from them. However, it encourages the users to manually check the generated metadata information for accuracy and encourages users to make improvements on it. Guided from these experience reports we decided to let DSPs input metadata information for their datasets, guided by the semantic support available in OnC. Towards this purpose, the PrC component has the interface PUI. Service providers input metadata information through PUI for every dataset they want to make available. Through the interface POI, PrC gets the semantic information from OnC for assisting dataset providers while constructing the metadata description for each dataset. So, for the metadata descriptions of all datasets under one (domain,sub-domain) pair, the PrC component creates one *Metadata Directory* (MTD) and one "local knowledge store" (for concept terms. Below, we give the key design features of PrC.

- *Formal MTD Structure:* There are different standards available on the web (*Schema.org, Dataset*, n.d.; Sordo, Tokachichu, Vitale, Maviglia, & Rocha, 2017; *Standard: Metadata*, n.d.) which dataset providers can follow in describing the functional, non-functional, and contextual contracts of datasets. Following the recommended guidelines in (Koesten et al., 2020) we have created a table of elements in Table 2.1 (Chapter 2). These elements are needed for a data seeker to understand datasets. So, we decided to set our standards using this list for describing the metadata. Another important decision that we have made is to follow the service-oriented design principles (Ibrahim, 2012) to design each metadata description as a "service" provided by a dataset provider. The functional aspect of metadata is separated from the contract part of metadata. They are "loosely coupled". So, we can change each one independent of the other. Consequently, one dataset functionality can be associated with different contracts. This way a dataset can be delivered to a variety of dataset seekers distributed geographically and bound by different sets of contextual constraints. So, we claim the PrC design is both new and novel.

- *Wizard-Based Metadata Publication:* The reviewed experience reports (Koesten et al., 2020; Strozyna et al., 2018) have used domain experts or data practitioners to create metadata for datasets. Following this guidance, in this thesis a "wizard" is conceptualized as OnC that provides "expert help" to PrC in preparing metadata descriptions of their datasets for publication.

- *Attribute Description:* Attribute section is part of the functional part of the service model of metadata. For each attribute, a tag (semantic term) chosen by the dataset provider from the ontology is associated. This will allow semantically correct search methods based on attributes (Koutras et al., 2021; Noy et al., 2019).

- *Quality Dimensions:* A table of quality dimensions is part of PrC. For each quality dimension, its type and unit of measurement are specified. A subset of the quality dimensions listed in this table is made mandatory. However, dataset providers can select additional set of quality attributes from this table and assign to the metadata description of each dataset.

- *Context:* The need to state the contextual information in data collection, and the purpose for which data is collected, are emphasized by different studies (Chapman et al., 2020; Koutras et al., 2021; Noy et al., 2019). However, they have not suggested a template or model for doing it. We use the context model (Wan, Alagar, & Pacquet, 2005) to describe context information. The importance of contracts in data discovery is highlighted in (Noy et al., 2019), but not implemented by any existing method.

3.4 Analysis Component (AnC)

The functionalities of ANT are encapsulated into this component. This component includes a rich interface AUI for analysts to construct queries based on GQM. The analysts can specify functional requirements with quality dimensions and semantic preferences in a query. Analysts consult the "local knowledge" corresponding to the (domain,sub-domain) pair of their interest in PrC component to construct semantically correct queries. Constructed queries are sent to the CeC component. In response to a submitted query, the CeC component will send to AnC the ranked list of metadata descriptions. Using the links in metadata descriptions, an analyst can access the datasets corresponding to the list of metadata received, and further filter it using the contextual constraints. So, the key design element of AnC is the AUI for query construction. Below we give the key features of AUI design, and the factors that motivated its design.

- *Model-based Design:* Currently existing DSEs (Chapman et al., 2020; Noy et al., 2019; Strozyna et al., 2018) have user interfaces that have only limited functionalities. For example, in Google DSE (Noy et al., 2019) analysts cannot build a query structure, but use only the "screen slots" to input a few keywords or select a domain of interest. Although suggested in (Chapman et al., 2020; Noy et al., 2019) that "an analyst needs a model for building the request", no such model was proposed. In this thesis the *Goal Question Metrics (GQM)* (Basili, 1992; Basili & Weiss, 1984; V. Solingen et al., 2002) method, originally proposed as "A Methodology for Collecting Valid Software Engineering Data", is used as the model in the background for interacting with analysts in every step of their way to construct query structures incrementally. The motivation to build this model-based AUI comes mainly from the need to construct a rich query structure.

- *Query Features and Rich Structure:* The information collected using the GQM model may be grouped under *Filters*, *Context*, *Attributes and Types*, *Quality Dimensions*, *User Semantics* and *Preferences*. Domain name, sub-domain name, and attributes as concept terms in a query are filters to retrieve "matching" metadata sets in CeC. Quality dimensions, user semantics, and preferences in a query are essential for "similarity-based" ranking. The above features are packed in a query structure that can be conceptualized as a vector or record with heterogeneous types. The model-based query construction process is made flexible and iterative to let the analysts cycle through their requirements until reaching a level of satisfaction to achieve "completeness", a subjective analyst-centric notion.

3.5 The Coupling Engine Component (CeC)

This component encapsulates the activities of DSE. It provides the "contextual tight coupling" between AnC and PrC components. The information contained in the analyser query structure and the information included in the MTD specific to the (domain,sub-domain) information provided in the query are used by this component in "matching" and "ranking" metadata sets. The two phases of CeC design are explained below.

- *Matching Phase:* Based on (domain,sub-domain) information in the query structure received from AnC, the metadata models in the MTD specific to that pair are first selected. Next, an algorithm based on the basic principles of *Formal Concept Analysis* (FCA) (Ganter & Wille, 2012) is used to extract the subset of metadata models that semantically match the input tagset.

- *Ranking Phase:* The selected metadata set and quality dimensions, user semantics, and preferences from the analyst query structure are input to a similarity matching algorithm. The algorithm calculates a similarity measure between every metadata model and the query input. Using these measures, the selected set of metadata set is ranked and sent to AnC.

3.6 Interfaces and Interactive Behavior of Components

A component can have many variables of different types defined internally. These are used for internal computations. Constraints for data access and data delivery can be expressed as logical expressions over these variables.

A component interacts with its external world through well-defined interfaces. In general, a component has two interface types:

- *User Interface*: This is a "front-end" of the component, through which users can interactively input and receive information. Many large systems have "graphical user interfaces", The interfaces AUI of AnC and PUI of PrC are of these types. Their input/output behavior are described as part of component description.

- *Component Interface*: This interface is one through which a component communicates with another component. Interfaces OPI of OnC, interfaces POI, PCI, and PAI of

PrC, interfaces API and ACI of AnC and interfaces CAI and CPI of Cec are of this type. These are shown in Figure 3.1.

A component interface is specified by a set of events occurring at it, where an event may have zero or more parameters. An event can be either an “input” event or an “output” event. Following the terminology for communicating components (Mohammad & Alagar, 2012), we denote an “input event x at an interface” as $x?$ and we denote an “output event y at an interface” as $y!$. An interface X of a component C_1 is *compatible* with an interface Y of component C_2 if every output event at X is an input event at Y, and vice versa. So, when $x!$ that is output at an interface of a component is received at the compatible interface of another component it becomes an input and hence we use the notation $x?$ for that message. In Figure 3.1, the interface pairs (POI,OPI), (PCI,CPI), (PAI,API), and (ACI,CAI) are compatible.

In specifying an interface behavior we include one or more constraints. A constraint of event $x!$ occurring at an interface is a logical expression involving the parameters of the event and the variables that are local to the component that outputs it. A constraint of event $x?$ occurring at an interface is a logical expression involving the parameters of the event and the variables that are local to the component that receives it. The constraints at each interface specification must be satisfied for the respective events to be executed. In turn, their successful executions formally describe the correctness of the internal functionality of the component. We use the above convention in later chapters.

To understand the interactive behavior of all the TCF components, assume that OnC is initialized first with all domain, sub-domains, and the ontology for every (domain,sub-domain) pair. Through PUI, the PrC registers dataset providers who want to make their datasets available. Then, it guides every registered dataset provider to construct the meta-data directories and local knowledge corresponding to the datasets they have. Next, the AnC component enables query construction with PrC interaction. Then AnC composes the query structure and communicates the constructed query structure to CeC component. The CeC component interacts with PrC component to match and rank the dataset links, and send them to AnC component. Finally, the analyst executes the contextual constraints in each metadata description of the ranked list received to filter out those that match the context at which datasets are actually requested for delivery. Thus, informally using interface connections shown in Figure 3.1, we get an overview of "interactive behavior" of the components in achieving the goal. So, with a detailed design and an implementation of each component design that is faithful to each interface behavior, the TCF will achieve its goal of finding datasets that are relevant to an analyst query. These are discussed in later chapters.

3.7 Merits of the Component Design

Component-based design has many advantages (Heineman & Councill, 2001; Mohammad & Alagar, 2012), including *simplicity*, *formality*, *generality*, *extensibility*, and *reusability*. The TCF design has these advantages. Formalization of the metadata descriptions and FCA-based searching ensure the retrieval process to be semantically correct. Generality is achieved because a variety of heterogeneous types are inherent in metadata information. The design has the external extensibility feature, because more components can be added through interface connectors, and has internal extensibility feature because an update of one component can be done independent from updates to other components. Reuse of TCF components is possible in other related service based projects, some of which we comment in Chapter 8.

In addition to the above advantages, the TCF design has some merits of its own.

- The *model-based* query construction is rich to include user semantics and preferences.
- The "service model" of metadata allows a loose coupling between dataset description and the context information. Provider context and service delivery context can be used by analysts to take delivery of datasets whenever and wherever analyst contexts match. So, *Contextual Closure* is enforced in the TCF design.
- The basis of matching method in CeC is FCA. Based on semantic correctness it retrieves metadata information that matches the concept terms in user query. Hence, there is no "ambiguity" in matching, and "every metadata" that is semantically matched will be retrieved.
- From the above three it follows that the retrieval of datasets is *complete*, in the sense

that “every dataset that is semantically relevant” will be sent to the analyst who can “filter out those that are irrelevant to the local context”. Moreover, the design is also *sound*, in the sense that “there is no junk” in the final result.

None of the DSEs that exist currently (Castelo et al., 2021; Noy et al., 2019; Strozyna et al., 2018) offers such an environment for dataset search. In Chapter 7, we will validate the above merits in the prototype case study. Further in Chapter 8 we use the above merits as criteria for comparing our dataset search engine with other available engines.

Chapter 4

Detailed Design of PrC

In this chapter we discuss a detailed design of the PrC component, after giving a brief explanation of the structure of component OnC, the "expert support" of PrC. Because "expert world" activities with OnC are outside the scope of this thesis and OnC is not implemented, we do not discuss OnC interface to experts. However, for the sake of clarity and comprehension of PrC structure and functionality, we decided to explain the support role of OnC to PrC in Section 4.1 and give the specifications of interfaces OPI and POI through which OnC and PrC interact in Section 4.4.

PrC design, shown in Figure 4.1, serves as the *Registry* in which user information, domain/subdomain names, and set of all metadata descriptions are stored. To provide services based on these stored information, it has the three sub-components *Authentication Component* (AUC), *Metadata Directory Component* (MTDC), and *Local Knowledge Component* (LKC). These component structures and their functionalities are discussed in Section 4.2.

The interface PUI of PrC is at the "edge" of PrC, for communication with dataset providers. Through PUI, service providers can register, and input metadata descriptions corresponding to the datasets they have. The datasets themselves are not input to PrC, but links to them are included as part of metadata information. Interface PUI is described in Section 4.3.3. The interfaces of PrC with AnC and CeC are explained in later sections.

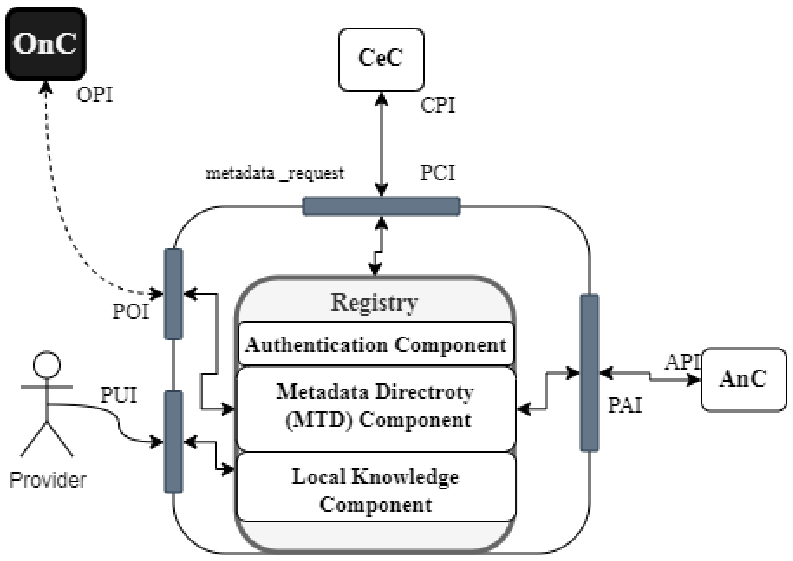


Figure 4.1: Design of PrC Component

4.1 Ontology Component (OnC)

Many studies that we reviewed in Chapter 2 have emphasized that terms used in metadata summaries must be supported by domain semantics in order to assure precision, clarity, and understandability. Many researchers (Chandrasekaran & Mago, 2021; Pawar & Mago, 2018; Seng et al., 2021; Trojahn et al., 2021) in ontology community believe that ontology must be specific to each domain/subdomain pair in order to improve comprehension and understandability of the semantics of textual summaries. Given that each domain will have its own specific knowledge repository, and a vast number of domains of interest exist for data practitioners, in this thesis we can only make certain assumptions on ontology structure in order to explain the support role of OnC to PrC. However, we will use a specific ontology for the concrete case study in Chapter 7. In this spirit we make the following assumptions.

(1) OnC is a black box and it is assumed to include an ontology for every (domain,subdomain) pair of interest for all dataset providers.

(2) Domain experts construct the ontologies. All key concept terms necessary to describe metadata information of one specific dataset within one (domain,subdomain) is included in one ontology. So, OnC is a component that includes expert knowledge from which semantically equivalent concept terms necessary to understand metadata descriptions can be recovered.

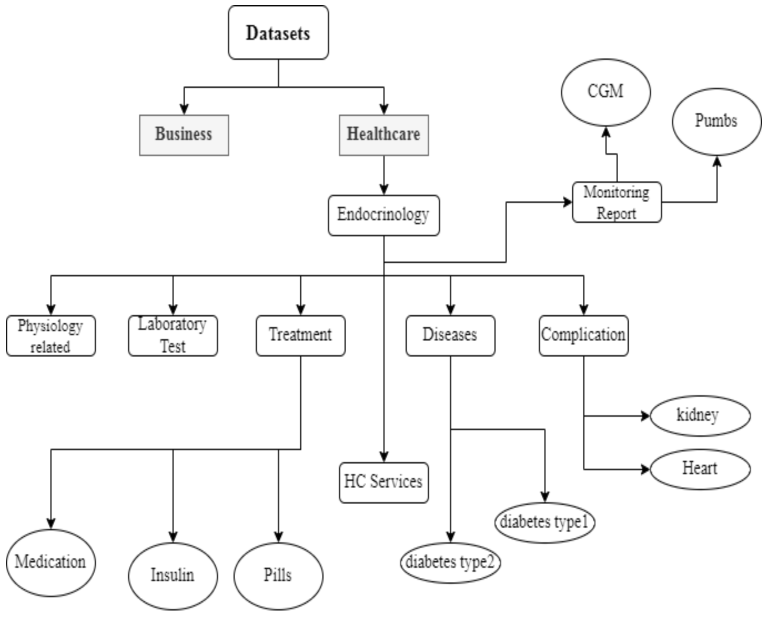


Figure 4.2: Conceptual Structure of Ontology Hierarchy

(3) An ontology structure is in general a rooted digraph, although for most applications it is a directed tree, as shown in Figure 4.2. The root of the tree is a *concept category* whose name is enclosed in a "rectangular box". Every sub-tree root of the root is a *sub-category* with the name of sub-category concept shown inside a rectangular box. All leaf nodes shown in "elliptical boxes" are "concept terms" under the sub-categories. The root of the smallest tree that includes two leaf nodes is the "closest category" to which these concept terms belong. In this thesis, "tags" are concept terms at the leaf nodes of the ontology included in OnC.

4.2 Sub-components of PrC

In this section we explain the functionalities of the services provided by the three components AUC, MTDC, and LKC, and the data structures needed locally in each component to fulfill the service functionality.

Figure 4.3 conceptualizes the structure of metadata directories and their corresponding local knowledge for one domain. So, for a given set of domains and a set of sub-domains under each one domain, a forest of trees structured as in Figure 4.3 will exist. Based on this conceptualization the two sub-components MTDC and LKC of the registry are constructed.

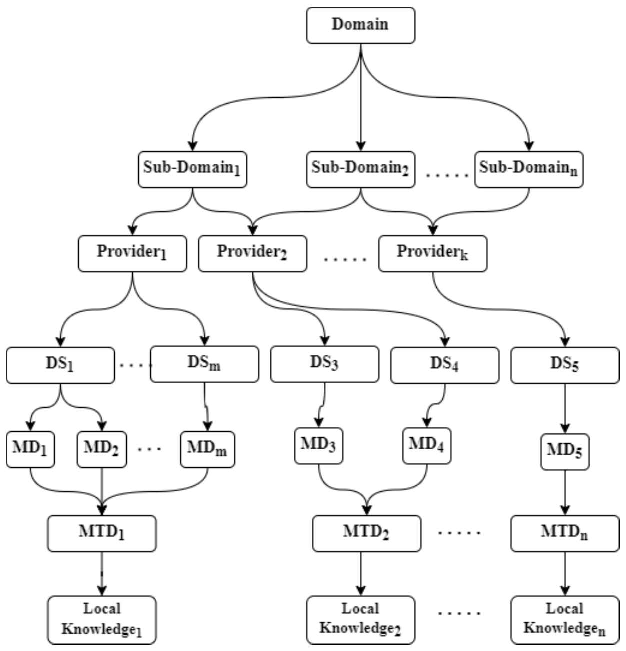


Figure 4.3: Conceptual Structure of PrC Registry

4.2.1 Authentication Component (AUC)

The only service provided by this component is initial registration and subsequent authentication of TCF users. In order to fulfill this service functionality, it has the local data structure *userSet*. Each element of this set is a triple (a, b, c), where a is the name of a user, b is the password chosen by the user to enter the site of TCF, and c is the user Id generated by the PrC for the user. We use the convention from abstract data type definition that *First*, *Second*, and *Third* are functions that respectively extract the first, second, and third components of a triple. So, if $x \in userSet, x = (uname, upassword, user\_id)$ then $First(x) = uname$, $Second(x) = upassword$, and $Third(x) = user\_id$.

Analysts will use the interface API and service providers will use the interface PUI for requesting initial registration and subsequent authentication. AUC will be commanded by PrC to fulfill these requests, and PrC sends the services through the interface PAI. The specification of the request from the analyst and the specification of PrC response to that request are given in Chapter 5.

4.2.2 Metadata Directory Component (MTDC)

As shown in Figure 4.3, one MTD exists for each (domain,sub-domain) pair. An MTD holds the set of metadata descriptions of all datasets available within that (domain,sub-domain) pair. The service functionality of MTDC is to manage the set of all MTDs, and provide services for uploading the metadata descriptions, and for accessing the information in it for all authenticated users. In order to fulfill these services, MTDC has a set of local variables as explained below.

- *Domain Names*: The domain names are stored in the variable $domSet$ of set type.
- *Sub-domain Names*: For each $d \in domSet$, the sub-domain names of d are stored in the variable $subdomSet(d)$ of set type.
- *Metadata Index for MTD*: MI is a two-dimensional array whose elements are of type sets. For $d \in domSet$, and $sd \in subdomSet(d)$, $MI[d, sd]$ stores the set of metadata descriptions of all datasets belonging to the pair (d, sd).
- *Metadata Identifier:* For every element $x \in MI(d, sd)$, the variable $mdid(x)$ denotes the unique identifier of the metadata description.

The metadata description format, as emphasized in (Koesten et al., 2020), is the key aspect of MTD design. We model a metadata description template as a *service*. Because of its importance we give an elaborate treatment of it separately in Section 4.3.

4.2.3 Local Knowledge Component (LKC)

The service functionality of this component is to assist analysts with semantic information of metadata to construct semantically meaningful queries. In order to fulfill this service, this component will search the array MI in PrC for a given (d, sd) pair. From each metadata model x stored in $MI[d, sd]$, it collects the set of tags and computes the set $tagSet(d, sd)$ as

$$tagSet(d, sd) = \bigcup_{x \in MI[d,sd]} tags(x),$$

where the set $tag(x)$ is the set of tags in the attributes part of the metadata model x. The set $tagSet$ is the "Local Knowledge" for all metadata models constructed by the service providers for the domain/sub-domain pair (d, sd).

4.3 Service Model of Metadata

The primary motivation to choose "service model" to describe metadata comes from the fact that there is an entity who "provides data services" and there is an entity "requesting data for a specific purpose", and a "contract is required" to provide data services to data seeker. In this paradigm, the service provider needs to describe "service functionality", "quality of product", and "contextual contractual information" for service delivery. The service seeker can "search for services that match the goals", and get those that fulfill "quality and context criteria" of the seeker. From the literature, we found the service model introduced by (Ibrahim, 2012) to be generic and flexible enough to include elements and features of metadata descriptions suggested by (Koesten et al., 2020; Strozyna et al., 2018). After examining our requirements for metadata summaries, we found that the above suggestions can be fitted into the previous service model (Ibrahim, 2012) with a few adaptations. This adapted model is shown in Figure 4.4.

Conceptually, the service model includes two parts, called Service and Contract. In the service part the semantic part of a dataset is described. This includes functionality and non-functional aspects of service. Contract part describes the contextual constraints governing dataset provider, data source, and data delivery. Because of the loose coupling between these two parts, one part can be changed independently from the other. Consequently, contract part may be changed without changing datasets functional part in order to offer it to a different set of seekers. In the following sections we explain the different parts of the service and bring out the expressive power of the model to accommodate a variety metadata descriptions.

4.3.1 Service

The "Functional" and "Non-Functional" parts explain the core information of metadata. Below, we discuss them in detail.

- **Functional:** The function has a "name" that suggests what the "service function

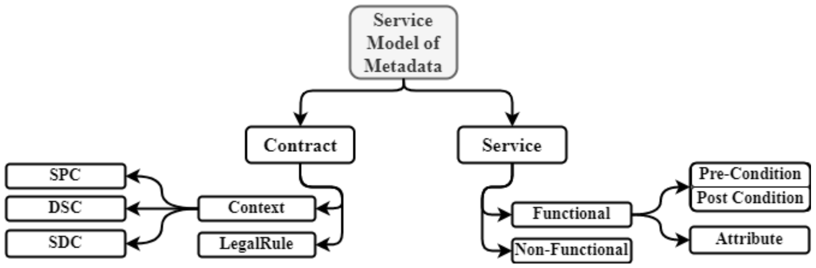


Figure 4.4: Service Model of Metadata

does". Unlike in programming where a function may include arguments, in the modeling level we let the function name to be simple. The "Attribute" part and the non-functional part explain the traits and qualitative characteristics of the function. Examples of suggestive function names in our model are "diabetic-data", "Covid-data", and "retinopathy-data". As confirmed earlier, in the thesis only open source data (either with or without registration) is considered. As such, the pre-condition to be met by users to get access to the metadata and the dataset described by it is kept simple. For free datasets, a user may be asked to enter "name, affiliation, and contact e-mail". For datasets that require "pre-registration", the user will be authenticated through the "user-id" issued at registration time. To be consistent with the convention that every function should return a result (value), the postcondition in our model will include propositional constants such as "granted" to denote the value of function for successful authentication. In addition, it may inform through predicates in the post-condition the confidentiality policy of collected information and how it might be used internally to enhance service management.

The term "attribute" in the context of mathematics means the "traits or the properties" of an object. In particular, a function definition in a program will include a set of attributes for introducing identifiers (parameters), inputs, and outputs. Given that data from large datasets are used in functional analysis, attribute-based descriptions of data summaries help data seekers understand the usability of datasets in their analysis. Many experiments reported in (Koesten et al., 2020) confirm that dataset seekers demand a well-chosen set of attribute names to describe the metadata summary in order to increase their understandability of the datasets behind the metadata. Following this guideline, we include in the service model "Attribute Part".

The elements that describe an attribute have the following significance.

- ◦ “Name” is the attribute name chosen by the service provide. Typical examples are “Disease”, “Age”, and “Gender”. An attribute is associated with a type.
- ◦ “Tag” is the concept term chosen by the service provider from the local knowledge associated with the (domain,sub-domain) subtree to which the metadata belongs. So, the tag associated with attribute name provides the “semantic” context for the name.
- ◦ “Type” describes the data type of the attribute. The type of data can be either a primitive type (numeric, string, boolean, array) or compound (record, tree). Type information will make precise the “value domain” and the “set of operations” defined for that domain.
- ◦ “isKey” element specifies whether or not this attribute is the key of a dataset record.
- ◦ “Description” element provides essential syntactic/semantic details about the attribute content. As examples, “Description” for a numerical attribute could be “the unit of measurement for values of that attribute or formula used to calculate the value”, and “Description” for “Date” attribute can include the format for specifying it.
- ◦ “Completeness” element states a percentage (a statistical measure) for the data content for this attribute. As an example, for the attribute “Ethnicity”, if the value given is 80%, then it means that 20% of data do not have a value for this attribute. So, “Completeness” information helps the analysts understand the proportion of missing values for the attribute.

- **Non-Functional:** Non-Functional section in the service model serves a different purpose from the “attributes section” of service. Attributes are related to “metadata variables”, whereas, non-functional part answer questions related to “goodness and suitability of dataset”. In general, non-functional properties include domain/subdomain information, a precise title to indicate the dataset content, and a set of quality features. We aim to provide an informative representation of the non-functional part to help requesters understand the dataset. So, we decided to make non-functional part of the metadata model “semi-schematic”. In it, a set of non-functional parameters are pre-defined and made mandatory for the service provider to include them in the metadata description. Nevertheless, the provider is given flexibility to add other descriptive elements chosen from Table 4.1 which is extracted from a comprehensive list non-functional items suggested by previous studies (Bian et al., 2020; Koesten et al., 2020; Pipino et al., 2002).

Table 4.1: Additional Non-functional Elements

Element	Description
Documentation	Documentation file of dataset collection
References	References to other datasets or data sources
Language	Language used to present the content of dataset
Cleanliness	Level of standardizing and removal of errors
Methodology	Describes the methods used to collect the data

Below, we briefly describe the meanings of elements that describe the non-functional part:

- Title: This element provides a general statement describing the metadata content. For example, “Breast Cancer Statistics for 2022” can be a title for a metadata. The title is an important element needed by the analysts to get a general idea of what the metadata is about Koesten et al. (2020). The type of Title element is String.
- Description: This element is to describe the metadata in some detail. It is a script where provider can give a lengthy description or give a set of keywords.
- Data Source: This element states the source from which the data is collected, and makes clear whether it is a primary or secondary data source. This element is important for dataset users to make a decision.
- Quality Dimensions: A variety of quality dimensions are mentioned in literature (Alsaig, Alagar, & Ormandjieva, 2018a; Koesten et al., 2020; Pipino et al., 2002; Strozyna et al., 2018). Below, we list them with their suggested meanings, explain the reasons for not including some in our model, and how it is enforced if included in our model.

Table 4.2: List of Quality Dimension Provided to Providers

Quality Dimension	**Data Type**	**Value-Measurement**	**Required?**
Volume	Numerical	gigabyte	Mandatory
Variety)	Nominal	{xml,cvs,text, ..}	Mandatory
Velocity	Numerical	Number of days	Mandatory
Release Year	Enumerate	{1995,....,2050}	Mandatory
Availability	Interval	[2021,2023]	Optional
Reliability	Categorical	{low, medium, high}	Optional
Safety	Categorical	{low, medium, high}	Optional
Veracity	Categorical	{not known, acceptable, verified}	Mandatory
Value*	Numerical	Percentage	Mandatory

∗ Accessibility: It is assumed to exist for all registered users once metadata description is uploaded in PrC by a service provider. So, we do not include it in our model.

∗ Availability: It gives the "duration", the number of years the dataset will be available. It is of type Interval. An example is $[2021, 2024]$.

∗ Relevance: The CeC component uses semantic based matching for selecting datasets that match the semantic tags in a query. So, the selected metadata descriptions are "semantically relevant" to the user. Analysts will be validating contextual contracts in the ranked metadata before selecting datasets for delivery. That is, only those datasets that are "contextually relevant" will be obtained by analysts. Because of these enforcement by design, we do not need to explicitly include relevance as a quality dimension.

∗ Format: This refers to heterogeneity of data/information. Our service model uses a variety of formats. Attributes and tags of attributes that provide semantic support are associated with type information. Hence, format for functional part is very rich. Contexts and legal information have their own formats. So, different format types are made part of our design.

∗ Consistent Representation: Every attribute of a certain type is fixed in the system. All metadata descriptions conform to one service model and all local knowledge conforms to one specific model. Therefore, consistent representation is enforced at both syntactic and semantic levels in our design.

* Completeness: This element refers to the “percentage of missing values” of an attribute of the dataset. It is included in every attribute representation. Completeness has numerical data type.
* Volume/(Data Size): This element describes the dataset size available for a metadata. In our model the size is a numerical variable that has a fixed unit of measurement which is “gigabyte”. A value for this element should be provided by dataset providers.
* Variety: This element states the file format in which the provider will provide the data. This variable is structured to be categorical in our model and it is mandatory to be provided by the provider.
* Velocity/(Update Frequency, Freshness): This element indicates the frequency of dataset update in its source. For instance, an update frequency can be “every day”, “every week”, or “every year”. This element is mandatory to be provided by the provider. In our model, the type of this dimension is “Numerical” and fixed to have “Number of Days” as unit of measurement.
* Release Year: This element refers to the year when data in a dataset was collected and started to become available. The type of this element is “enumerate”. That is, the dataset provider chooses one value from the enumerated values and assigns it for the year of dataset release.
* Safety: If a decision made by an analysis of data from the dataset of a metadata description leads to unsafe situation, then we can say “data is not safe”. We expect providers use this dimension to state the safety levels for using the data. Here, safety refers to “work place safety” where the dataset content will be used. The type is Categorical (ranked) with values (low, medium, high). They may be mapped to numerical values, such as (low = 1, medium = 3, high = 5).
* Reliability: It implies the “level authenticity of data”. This element is of type Categorical (ranked) with values (low = 1, medium = 3, high = 5). “Low” indicates *high level of fake/augmented data*, and “high” indicates *data collected from real-world*.
* Veracity/(Accuracy): This dimension reals the “level of truthfulness” of

data. This element is of type Categorical (ranked) with values (not known, accepted, verified). They can be mapped to numerical values as in (not known = 1, accepted = 3, verified = 5).

∗ Value: The meaning of this quality dimension is "value-added" (in service paradigm), from the *user-centric* perspective. A service provider cannot assign a "value" to "Value dimension". In our model, a "value" for the "Value dimension" is automatically computed by CeC component on every metadata selected by it after the matching and selection phase. We explain this calculation in Chapter 6.

We use type to describe the value domain of quality dimensions. The type of data can be either a primitive type (numeric, string, boolean, array) or compound (record, tree). Type information will make precise the "value domain" and the "set of operations" defined for that domain. In our model, we allow "Set Type", "Ranked Categorical Type", "Range Type", and mixed heterogeneous types such as "Set of Records". We choose the correct operations for each quality dimension during ranking process of metadata in CeC.

The subset of quality dimensions that we have chosen from those discussed above are listed in Table 4.2. These are for service providers to use in metadata descriptions. The ⋆ on "Value" dimension is to tell service providers that they should leave "Value" field "empty" in metadata descriptions. The last column indicates whether or not a quality dimension is mandatory. In our model, the mandatory ones are included in the metadata template as fixed required features. Service providers can add other optional (non-mandatory) quality dimensions listed in Table 4.2.

4.3.2 Contract

Every service has a contract. In the metadata service model, contract part has the two elements *Context*, and *Legal Rules*. The reason why context becomes important in a contract is that metadata dimensions such as "who", "when", and "where" that arise in service transactions are in fact "context dimensions" (Wan, 2006). As part of legal rules we include contextual constraints that bind service delivery to service providers and service requesters.

In a service provision, service provider, service requester, and service are the three entities. Each entity can be associated with a context. The Service Provider Context (SPC) includes information on “SP-NAME”, “SP-LOC” (for Location), and “SP-CONT” (for Contact). The context associated with the service, called Data Source Context (DSC), includes information on “DS-ORI” (for Origin) indicating the location from where the data in the dataset was collected, and “DS-RT” indicating the “release year” of the dataset, and “DS-URL” indicating the link to dataset. The context that the service provider creates for service requester can be Service Delivery Context (SDC). It has information on “SD-REG” indicating a set of regions where this service is not available, “SD-ORG” indicating a set of organizations to whom the service is not free, and “SD-FEE” indicating the license fee (not fee for data) for dataset downloads. So, in the contract part of service model these three context types are included in the context notation (Wan, 2006).

Example 1. *Examples of the three context types in this notation are given below:*
SPC:$[SP-NAME : ABCInc; ; SP-LOC : Toronto; SP-CONT : \{Address,\ e\text{-}mail\}]$
DSC:$[DS-ORI : XYZHospital; DS-RT : 2011; DS-URL : abcbigdata.com]$
SDC:$[SD-REG : \{country_1, \cdots, country_m\}; SD-ORG : \{Org_1, \cdots, Org_k\}; SD-FEE : 250]$

The contexts SPC and DSC will be examined by an analyst to make a decision on whether or not the dataset received from CeC is desirable. Once it is determined as desirable, the analyst will validate the local analyst context with SDC to ensure data delivery is possible in the local context. After completing these steps on the ranked list of metadata received by the analyst, the datasets filtered by contexts can be downloaded by the analyst. Thus, every dataset obtained by the analyst satisfies the semantic requirements in the analyst query, has the quality features that satisfy the preferences and similarity semantics specified in the query, and finally the contexts are compatible. In practice, every nation and almost every organization has developed and published its own set of legal rules for data access, data sharing, and data reuse. It is a wide area of importance. In this thesis, our goal is only to emphasize that “legal rules” are essential part of service contract part and give a few simple examples of legal rules. However, it is very hard to state them formally, as in legal terms, and have a method to validate them. The TCF system does not enforce/validate such rules. The analyst can review these rules before requesting service delivery. We explain the role of analyst to resolve contextual and legal issues in more detail in Chapter 5.

Some examples of legal rules are given below. Some rules have a specific context to apply.

- Data may not be redistributed or shared with third parties. This rule may be applied at the time of service delivery context.
- Dataset provider retains the right to amend data and users will be alerted when amendments are made.
- Contract renewals should be renegotiated. The context may be defined by "a duration of time" preceding the contract termination time.
- Data can be used for educational and research purposes.

4.3.3 Provider User Interface (PUI)

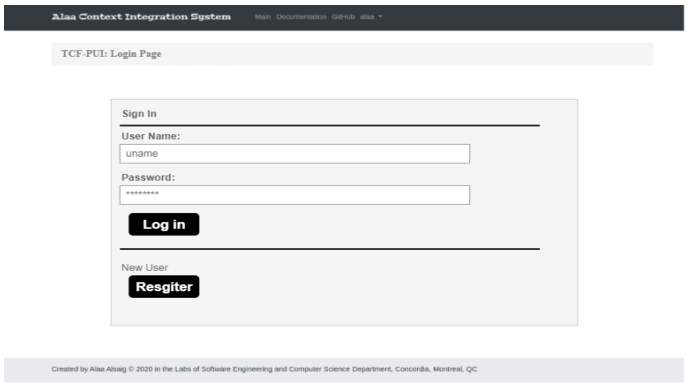
PUI is the front end "graphical user interface" which helps providers interact with PrC component and input information to build metadata service models. PUI design principles are as follows:

- *Separation of Concerns:* The different parts of metadata model can be independently constructed, reviewed, edited, and saved.
- *Separation of Files from Directory:* Files of completed "metadata parts" are saved separately in a private space assigned for the user. A file from this storage can be viewed, edited and saved again independent from other related files.
- *Uploading Metadata Descriptions:* A metadata description is obtained by linking the files (saved metadata parts) using the unique metadata identifier assigned to the saved parts. Only after linking, a metadata description is complete and can be posted in Metadata Directory (MTD). The MDT uses the metadata identifier to allocate directory space.
- *Read-only MTD:* The MDT has "read-only" privilege.

PUI guides the user through many interfaces to construct the metadata description. In addition to “Register” and “Login” buttons that enable user authentication, the “general buttons” in many interface windows serve the following purpose.

- *Cancel:* This button in “Registration” interface, will cancel the registration session and open the “Login Interface” window.
- *Exit:* This button enables the user to terminate the interaction session with PUI. The “Login” interface window opens upon exit.
- *Save:* This button enables the user to save the data input through that window.
- *Next:* PUI has its “defined sequence of steps”, enabling the user to input data for different parts of metadata. This button enables the user to view the “next” interactive window defined in PUI.
- *Back:* This button enables the user to return to the interactive window of the proceeding step.
- *View:* This button enables the user to return to “Metadata Storage” interface window, where the user can enable the special buttons in that interface to view saved files, link saved files, and upload the completed metadata descriptions.

There are other buttons that are specific to some windows. We will explain them in the following sequence of PUI steps.


Alaa Context Integration System
Main Documentation GitHub alaa
TCF-PUI: Login Page
Sign In
Log in
New User
Resgiter

User Name: uname

Password: ********

Created by Alaa Alsaig © 2020 in the Labs of Software Engineering and Computer Science Department, Concordia, Montreal, QC

Figure 4.5: Login Interface

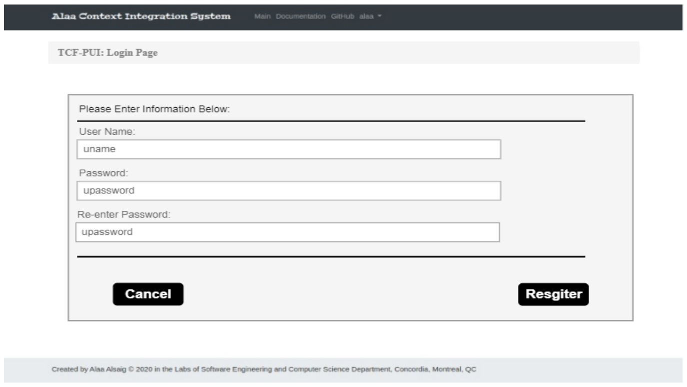

Alaa Context Integration System
Main Documentation GitHub alaa
TCF-PUI: Login Page
Please Enter Information Below:
User Name:
uname
Password:
upassword
Re-enter Password:
upassword
Cancel
Resgiter
Created by Alaa Alsaig © 2020 in the Labs of Software Engineering and Computer Science Department, Concordia, Montreal, QC

Figure 4.6: Registration Interface

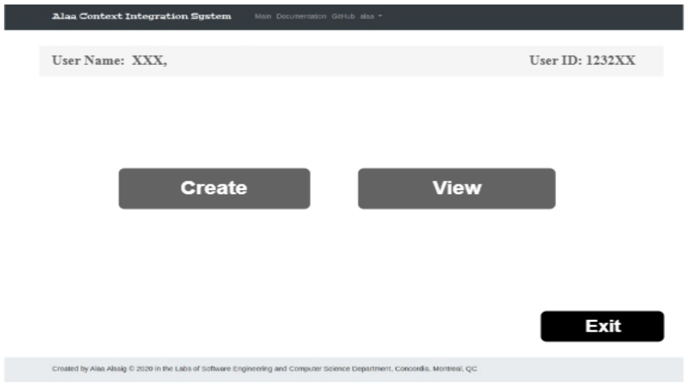

Alaa Context Integration System
Main Documentation GitHub alaa
User Name: XXX,
User ID: 1232XX
Create
View
Exit
Created by Alaa Alsaig © 2020 in the Labs of Software Engineering and Computer Science Department, Concordia, Montreal, QC

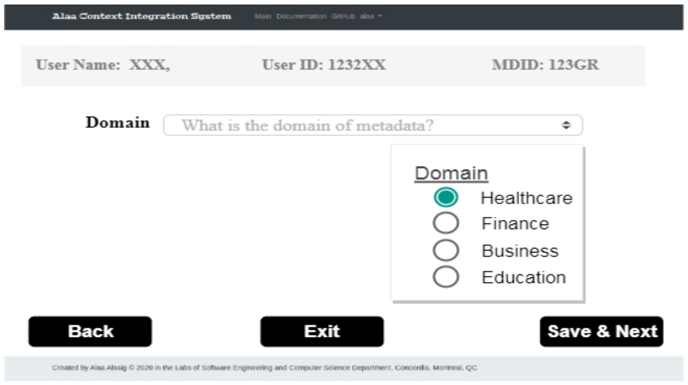
Figure 4.7: Metadata Startup Interface

- **Entry:** A user is presented with the "Login" interface window upon opening the PUI. If the user is already registered, the user will enter the $(username, password)$ pair. If the user is authenticated, then "Metadata Startup Interface" window in Figure 4.7 opens, with the assigned *user_id* of the user. If the user is not authenticated, either the user retries or closes the window to exit from PUI, or click on "Register" button. In the later case the "Registration" Interface window in Figure 4.6 opens.

- **Registration:** Initially, every service provider must register with $(username, password)$. The registration interface screen in Figure 4.6 serves this purpose. When a user is registered, the authentication unit of PrC component generates a *user_id* for the user. This identifier will be shown in all subsequent dialogue windows for this user. Upon successful registration, the "Metadata Startup Interface" window in Figure 4.7 opens.

- **Subsequent Authentication:** After exiting from a session using the "Exit" button in a window, the user will be shown the "Login" interface window. The user can start a new session after sign in or close the window to leave the PUI.

- **Creation of New Metadata and Viewing Existing Metadata:** Upon initial registration and after subsequent authentications the "Metadata Startup Interface" window in Figure 4.7 opens. To start a new metadata creation, the user clicks on the "Create" button in Figure 4.7, which opens up the *Domain* screen in Figure 4.8. To review and edit existing (saved) metadata descriptions, the user has to click on the "View" button in Figure 4.7. which opens up the screen in Figure 4.10.


Alaa Context Integration System
Main Documentation GitHub alaa

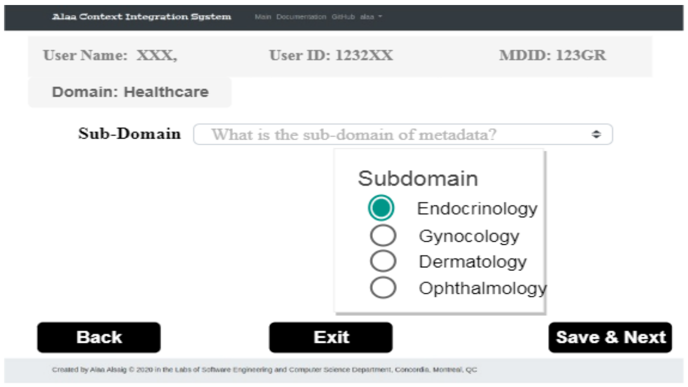
User Name: XXX, User ID: 1232XX MDID: 123GR

Domain: What is the domain of metadata?

Domain
[✓ Healthcare]
Finance
Business
Education

Created by Alaa Alsaig © 2020 in the Labs of Software Engineering and Computer Science Department, Concordia, Montreal, QC

Figure 4.8: Domain Interface


Alaa Context Integration System
Main Documentation GitHub alaa

User Name: XXX, User ID: 1232XX MDID: 123GR

Domain: Healthcare

Sub-Domain: What is the sub-domain of metadata?

Subdomain
[✓ Endocrinology]
Gynocology
Dermatology
Ophthalmology

Created by Alaa Alsaig © 2020 in the Labs of Software Engineering and Computer Science Department, Concordia, Montreal, QC

Figure 4.9: Subdomain Interface

The user might navigate two different ways across the dialog screens depending on whether the user wants to create a new metadata or view/edit an existing metadata file. We discuss these navigation separately below.

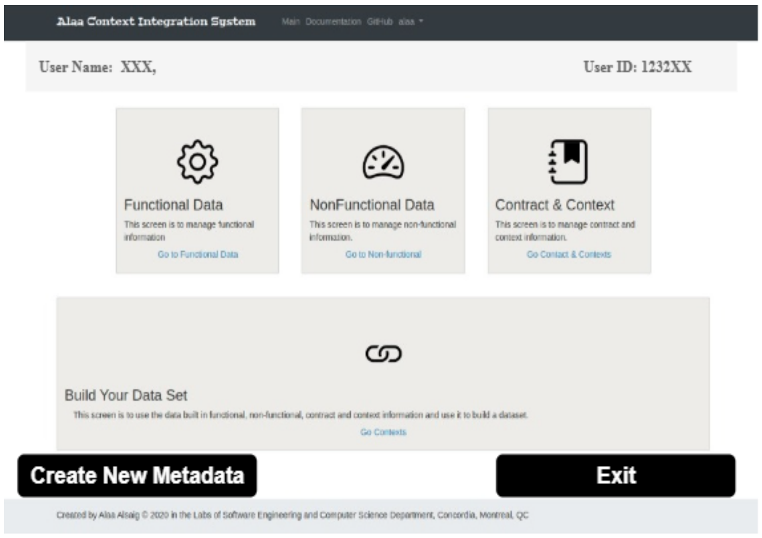
- **New Metadata Creation:** The system creates a new unique identifier *mdid* for the metadata to be created. This identifier will appear in all dialog windows, starting with domain creation window.
  - **Domain Interface:** As a first step for creating metadata, a service provider is required to select a domain name. The "Domain" text-box, when clicked, opens a list of domain names, as shown in Figure 4.8. In the background, the PrC communicates with OnC to fetch the set of domains. The formal specification of interface communications between PrC and OnC are given in Section 4.4. After selecting a domain name, the service provider has three options. Using "Back" button, the user can go back to the previous window and make a new decision, or the user can exit, or the user can click on "Next" button to move on to the sub-domain selection window. The domain name, stored temporarily, will be carried on and appear in all subsequent windows.
  - **Sub-Domain Interface:** The "Sub-Domain" tex-box , when clicked, opens a list of domain names, as shown in Figure 4.9. In the background, the PrC communicates with OnC to fetch the set of sub-domains for the selected domain. The formal specification of interface communications between PrC and OnC are given in Section 4.4. After selecting a sub-domain name, the service provider has three options. Using "Back" button, the user can go back to the previous window and make a new selection of dimension, or the user can exit, or the user can use "Next" button to move on to the next window for creating the functional part of the metadata. The sub-domain name, stored temporarily, will be carried on and appear in all subsequent windows.
  - **Functional, Non-Functional, and Contract parts:** The interfaces to create these parts are respectively shown in Figure 4.11, Figure 4.12, and Figure 4.13. Each window is designed to receive the information as discussed earlier. On each window, the (domain,subdomain) pair selected earlier, the *user_id*, and the metadata Id will become part of the metadata description. The significant interactions in each interface window are explained below.
    - **Functional Part:** When the interface button for inputting attribute

information is clicked, the user is presented with panels to input information on the six sub-fields in it. When “Tag” field is clicked, a panel with the set of tags retrieved from the ontology corresponding to the (domain,sub-domain) name that appears on this interface window. This retrieval, done in the background, is facilitated by the interface communications explained in Section 4.4. The user selects one tag for each attribute name entered. Input descriptions for rest of input buttons are straightforward. The user must click on “Save” before exercising the other options in order to have the functional part information saved in a file under the *user_id* and the *mdid*. This file is part of the storage allocated to all functional parts created in the system. Using the interface window in Figure 4.10, these files can be viewed and edited by the service provider who created it. If “Save” option is not exercised before exercising any other option, the input information will be lost. Clicking on “Next” button the user will open the “Non-Functional Part” interface screen. The other buttons have their meanings as explained earlier.

- **Non-Functional Part:** In the model, a set of non-functional parameters are pre-defined and made mandatory for the service provider to include them in the metadata description. Nevertheless, the provider is given flexibility to add other descriptive elements. A scroll-down panel will list these elements (see Table 4.1). When the interface button for inputting quality dimensions is clicked, the user is presented a scroll-down panel in which the quality elements from Table 4.2 are displayed. The user may select zero or more from this table. These selected elements and all elements marked with $\star$ will be added to the model. Input descriptions for rest of input buttons are straightforward. The user must click on “Save” before exercising the other options in order to have the non-functional part information saved in a file under the *user_id* and *mdid*. This file is part of the storage allocated to all non-functional parts created in the system. Using the interface window in Figure 4.10, these files can be viewed and edited by the service provider who created it. If “Save” option is not exercised before exercising any other option, the input information will be lost. Clicking on “Next” button the user will open the “Contract Part” screen. The other buttons have their meanings as explained earlier.

  - **Contract Part:** From a set of pre-defined legal rules, the user can select a legal rule. The option “other” is included in this list. When the user selects “other” option, a dialog box appears for the user to input a legal rule. In fact, the user can input zero or more legal rules. The button to create contexts, will guide the user with inputting the information for each dimension for each context type. The user is free not to include a context, however we do not give a provision for the user to add a new context type. The user must click on “Save” before exercising the other options in order to have the description saved in a file under the *user_id* and the *mdid*. This file is part of the storage allocated to all contract parts created in the system. Using the interface window in Figure 4.10, these files can be viewed and edited by the service provider who created it. If “Save” option is not exercised before exercising any other option, the input information will be lost. Clicking on “View” button the user will open the “Metadata Storage” interface screen shown in Figure 4.10. The other buttons have their meanings as explained earlier.

- **Metadata Storage (View,Edit Files):** The buttons “Functional Data”, “Non-Functional Data” and “Contract & Context” in the interface shown in Figure 4.10 enable the user to respectively view and edit metadata description files saved under the categories “Functional Part”, “Non-Functional Part”, and “Contextual Contract Part”. Every saved file in these categories can be independently edited, using the “Edit” button which is part of the saved file. Once the “Edit” button of a file is clicked, that file will open in its “corresponding interface window”. For example, if a file in “Functional Part” is clicked, it will open in the interface window shown in Figure 4.11. The user can edit on this window, and save the file. When the button “Publish Metadata” is clicked, a scroll-down window appears in which the identifiers of saved metadata files will appear. The user must select the metadata identifier


Alaa Context Integration System
Main Documentation GitHub alaa
Create New Metadata
Exit

User Name: XXX, User ID: 1232XX

Functional Data
This screen is to manage functional information
Go to Functional Data

NonFunctional Data
This screen is to manage non-functional information.
Go to Non-functional

Contract & Context
This screen is to manage contract and context information.
Go Contact & Contexts

Build Your Data Set
This screen is to use the data built in functional, non-functional, contract and context information and use it to build a dataset.
Go Contexts

Created by Alaa Alsaig © 2020 in the Labs of Software Engineering and Computer Science Department, Concordia, Montreal, QC

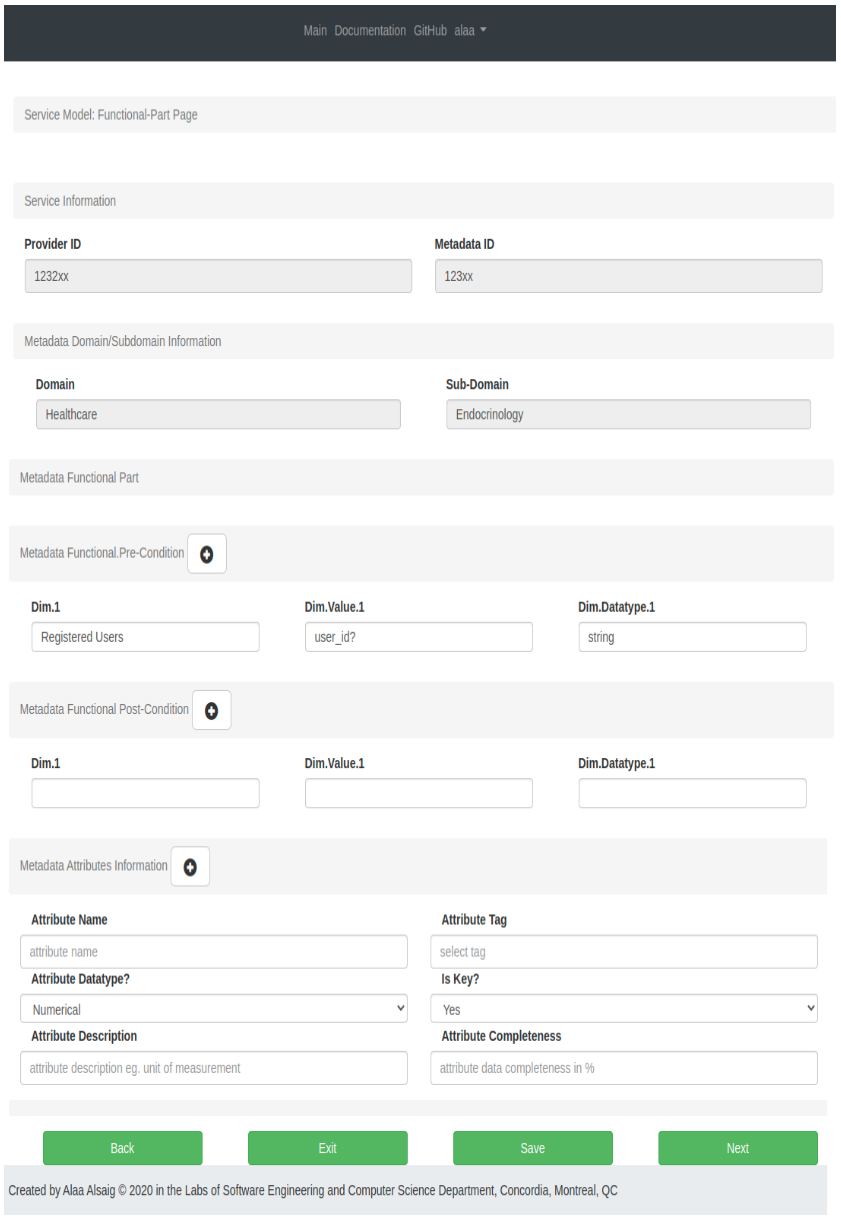
Figure 4.10: Metadata Storage- Access to View and Edit

that needs to be uploaded. The system will link the Functional, Non-Functional, and Contractual files that have the selected metadata identifier and upload it to the MTD in the registry.

Listing 4.1: Metadata Description in PrC MTD

```

{user_id: $Prov_1$,
mdid:$mdid_1$,
domain: "Healthcare",
subdomain: "Endocrinology",
md_functional:{
    precondition:[$cond_1$:regsiteredRequester=true],
    postcondition:[$cond_1$:requesterFeedback()=true],
    attributes:{
      names:['p_age', 'p_sex', 'lifestyle', 'food_consumption', 'alcohol', 'a1c'],
      tags:['patient age', 'patient gender', 'lifestyle patterns', 'consumption of
          food', 'alcohol', 'hba1c'],
      types:['string', 'string', 'string', 'numerical', 'boolean', 'numerical'],
      isKey:[0,0,0,0,0,0],
      description:["age in years", "", "", "in calories", "patient consumes
          alcohol or not", "3 month average of blood sugar"],
      completeness:[100,100,60,80,80,100]}},
nonfunctionalPart:{
    title:"diabetic patients life style",
   description:"projectxX in ddd",
   data_source:"proCompany",
   quality_features:{Volume:300, Variety:'cvs', Velocity:365, Release Year:2005,
       Availability: [2005,2015], Reliability:5, Safety:5, Veacity:0, Value:0.6}}
contextual_contract:{
    spc:[name:yang, SP.Location:"YY_MTL", sp.contact:"yang@email.com"],
    dsc:[origin:"Canada", releaseYear:2010, URL:"www.pro.com"],
    sdc:[region:"", sd-organization:"", SD-Fee:""
    legalRules: ["Data can be used for educational and research purposes."]
}}
```


Main Documentation GitHub alaa
Service Model: Functional-Part Page
Service Information

Provider ID: 1232xx

Metadata ID: 123xx

Metadata Domain/Subdomain Information

Domain: Healthcare

Sub-Domain: Endocrinology

Metadata Functional Part
Metadata Functional.Pre-Condition

Dim.1	Dim.Value.1	Dim.Datatype.1
Registered Users	user_id?	string

Metadata Functional Post-Condition

Dim.1	Dim.Value.1	Dim.Datatype.1

Metadata Attributes Information

Attribute Name: attribute name

Attribute Tag: select tag

Attribute Datatype?: Numerical

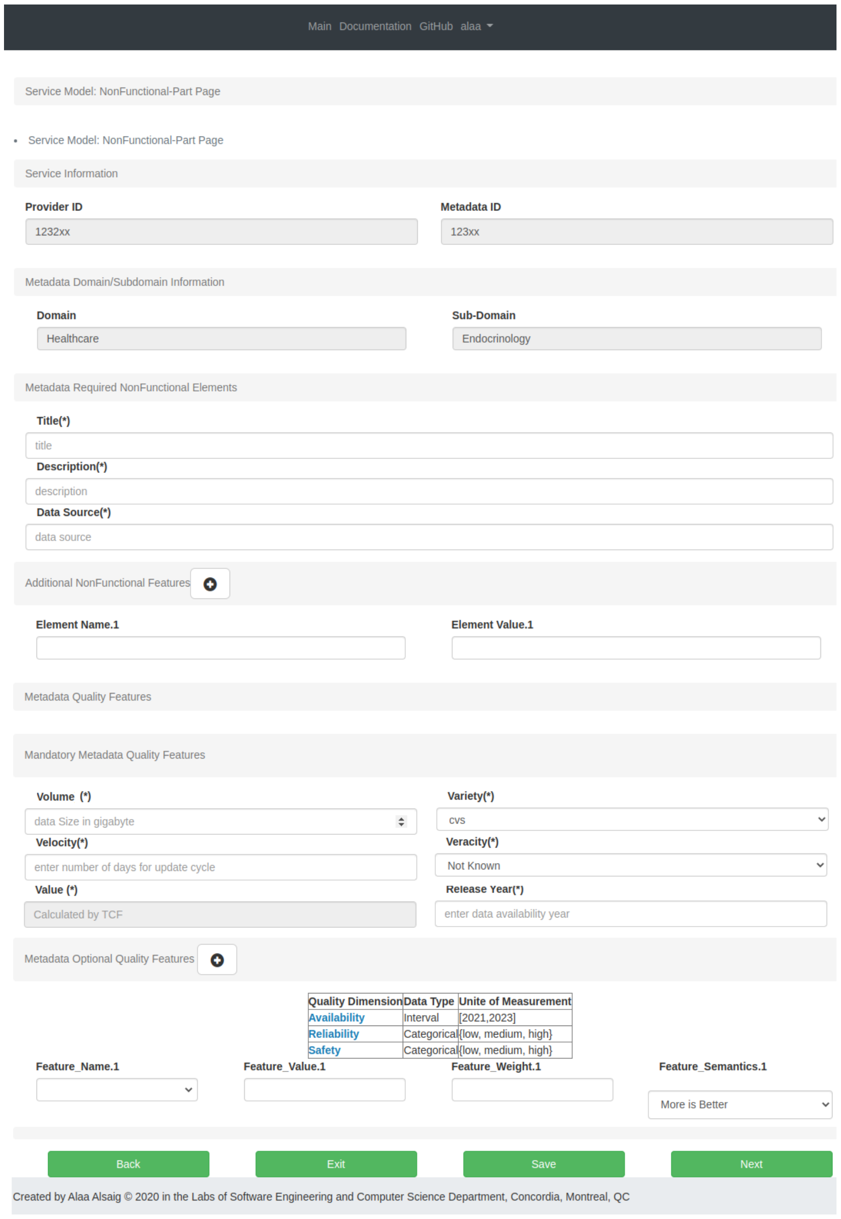
Is Key?: Yes

Attribute Description: attribute description eg. unit of measurement

Attribute Completeness: attribute data completeness in %

Created by Alaa Alsaig © 2020 in the Labs of Software Engineering and Computer Science Department, Concordia, Montreal, QC

Figure 4.11: Metadata Functional Part Page



Service Model: NonFunctional-Part Page

- Service Model: NonFunctional-Part Page

Service Information

Provider ID: 1232xx

Metadata ID: 123xx

Metadata Domain/Subdomain Information

Domain: Healthcare

Sub-Domain: Endocrinology

Metadata Required NonFunctional Elements

Title(*): title

Description(*): description

Data Source(*): data source

Additional NonFunctional Features

Element Name.1

Element Value.1

Metadata Quality Features

Mandatory Metadata Quality Features

Volume (*): data Size in gigabyte

Variety(*): cvs

Velocity(*): enter number of days for update cycle

Veracity(*): Not Known

Value (*): Calculated by TCF

Release Year(*): enter data availability year

Metadata Optional Quality Features

Quality Dimension	Data Type	Unite of Measurement
Availability	Interval	[2021,2023]
Reliability	Categorical	{low, medium, high}
Safety	Categorical	{low, medium, high}

Feature_Name.1

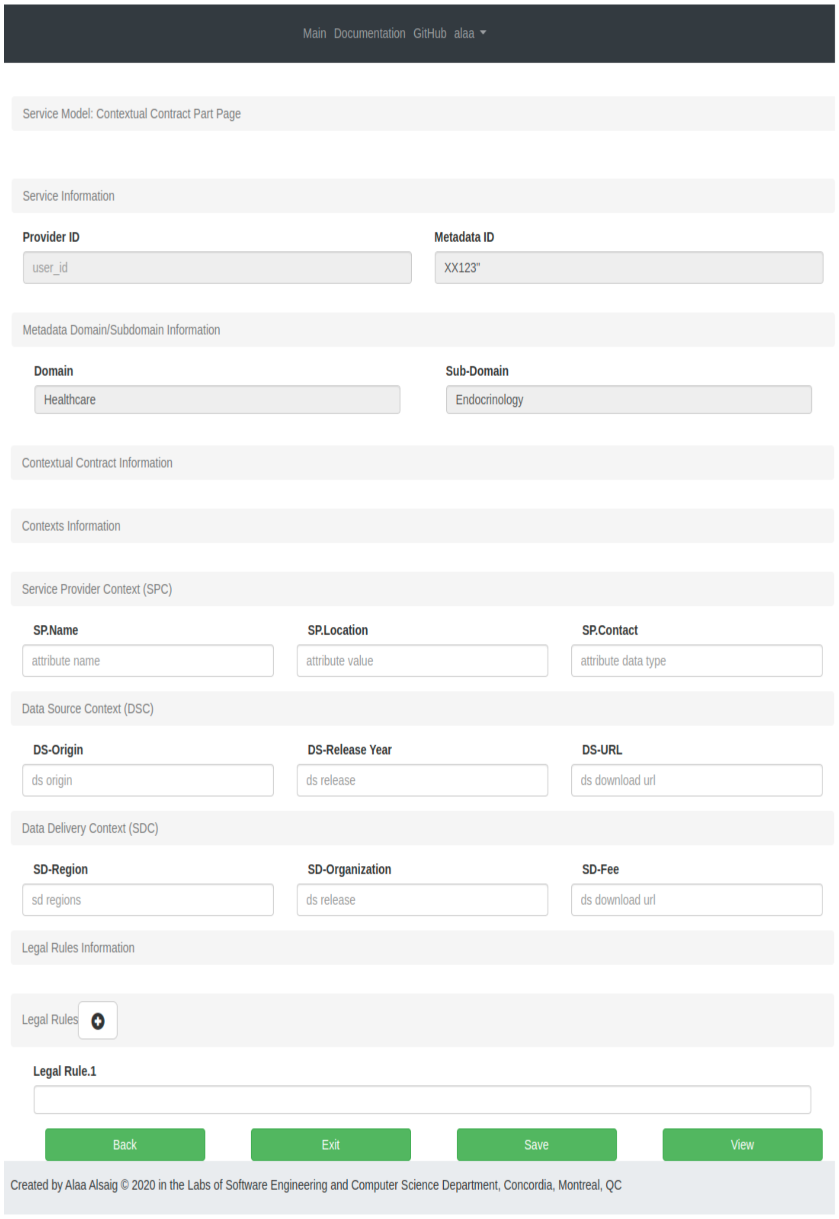
Feature_Value.1

Feature_Weight.1

Feature_Semantics.1: More is Better

Created by Alaa Alsaig © 2020 in the Labs of Software Engineering and Computer Science Department, Concordia, Montreal, QC

Figure 4.12: Metadata Non-Functional Part Page



Service Model: Contextual Contract Part Page

Service Information

Provider ID: user_id

Metadata ID: XX123"

Metadata Domain/Subdomain Information

Domain: Healthcare

Sub-Domain: Endocrinology

Contextual Contract Information

Contexts Information

Service Provider Context (SPC)

SP.Name: attribute name

SP.Location: attribute value

SP.Contact: attribute data type

Data Source Context (DSC)

DS-Origin: ds origin

DS-Release Year: ds release

DS-URL: ds download url

Data Delivery Context (SDC)

SD-Region: sd regions

SD-Organization: ds release

SD-Fee: ds download url

Legal Rules Information

Legal Rules

Legal Rule.1

Created by Alaa Alsaig © 2020 in the Labs of Software Engineering and Computer Science Department, Concordia, Montreal, QC

Figure 4.13: Metadata Contextual Contract Part Page

4.4 OnC Support to PrC: Interaction Between POI and OPI

In this section we explain the communications that happen in the "background" between the components PrC and OnC through the interfaces POI and OPI. We also use formal notations to emphasize the actions triggered by messages. In later chapters we explain many straightforward interface communications only informally.

The interface communications are triggered at "Domain Interface" (Figure 4.8), at "Sub-Domain Interface" (Figure 4.9), and at "Functional Part Page interface" (Figure 4.11). The variables defined locally in each component and a functional notation are used in specifying interface behaviors.

- **Domain Request:** In PUI, when the "Domain" text box is enabled in Figure 4.8, the PrC sends the (output) message $domain\_request(authorized\_id)!$ through POI to OnC. The parameter $authorized\_id$ is that of the user invoking the "Domain" text box. It is received at OPI of OnC as input message $domain\_request(authorized\_id)?$. The parameter is saved locally by OnC in its local variable $codeSet$ of type set. Formally, $codeSet = codeSet \cup \{authorized\_id\}$. OnC copies $domainSet$, the set of domains it has, in the parameter D and sends to PrC through the (output) message $domain\_send(authorized\_id, D)!$. This message is received by PrC as (input) message $domain\_send(authorized\_id, D)?$ at its interface POI, and it saves the set D in the local variable $domSet$ of type set. Formally, $domSet = D$. The elements of $domSet$ are then displayed in the scroll-down panel in "Domain interface" window.

- **SubDomain Request:** The communication behavior is quite similar to what we have explained for "Domain" request. So, we skip the details and show only message names, and the internal actions triggered by them.
  Message sent by PrC: $subdomain\_request(authorized\_id, d)!$. Here, PrC ensures that $d \in domSet$.
  Message received by OnC: $subdomain\_request(authorized\_id, d)?$ Here, OnC checks (validates) $authorized\_id \in codeSet \wedge d \in domainSet$.
  Message sent by OnC: $subdomain\_send(authorized\_id, d, SD)!$. Here SD is a copy of the set of subdomains of domain d in OnC. That is, $SD = subdomSet(d)$.
  Message received by PrC: $subdomain\_send(authorized\_id, d, SD)?$. PrC will copy

SD to its local variable $subdomSet(d)$. That is, $subdomSet(d) = SD$. The elements of *subdomSet* are then displayed in the scroll-down panel in "SubDomain interface" window.

- **Tag Set Request:** The communication for tag set request is enabled in the interface window in Figure 4.11). Below we show only message names, and the internal actions triggered by them.

  Message sent by PrC: $tags\_request(authorized\_id, d, sd)!$. In the "Functional part" interface window, the authorized user name, domain, and subdomain names all appear. These are copied as parameters in the message. That is, all parameters are valid.

  Message received by OnC: $tags\_request(authorized\_id, d, sd)?$. Here, OnC validates the parameters.

  $$authorized\_id \in codeSet \wedge d \in domainSet \wedge sd \in subdomainSet$$

  Message sent by OnC: $tags\_send(authorized\_id, d, sd, tagSet)!$. Here *tagSet* is a copy of the set of tags from the ontology in OnC for (d, sd) pair. That is, $tagSet = Tags(O) \wedge O = Ontology(d, sd)$.

  Message received by PrC: $tags\_send(authorized\_id, d, sd, tagSet)?$. PrC will copy *tagSet* to its local variable $tagSet(d, sd)$. That is, $tagSet(d, sd) = tagSet$. The elements of $tagSet(d, sd)$ are then displayed in the scroll-down panel in "Functional part interface" window.

Chapter 5

Detailed Design of AnC

This chapter discusses the detailed design of Analyst Component (AnC), which is dedicated for analysts to build queries and manage their requests. The functionalities of AnC are the following:

(1) Allow analysts get registered and authenticated in the system.

(2) Guide an authenticated analyst to construct queries through dialogues with PrC.

(3) Submit queries constructed by analysts to CeC.

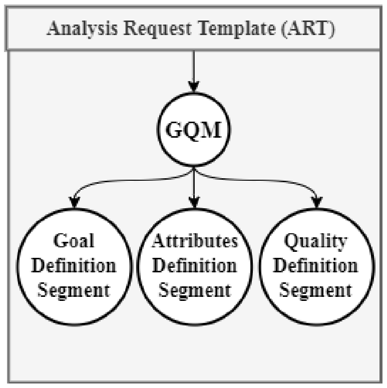
(4) For each submitted query, receive the ranked list of metadata descriptions that match the query.

(5) Enable the analyst to select from the received list those metadata descriptions in each of which the contextual contract description matches the analyst context.

AnC is designed to fulfill these functionalities through many AUI interface windows for direct interaction with analysts, through interface API for interaction with PrC, and through interface ACI for interaction with CeC. Below we explain how the above functionalities are fulfilled.

5.1 Registration and Authentication of Analysts

Every analyst must register first and be authenticated before using AnC facilities. The AUI provides registration and login interface windows for this purpose. These interface



Analysis Request Template (ART)
GQM
Goal Definition Segment
Attributes Definition Segment
Quality Definition Segment

Figure 5.1: ART: Abstract Model

windows are similar to those in PUI, defined in Chapter 4. So, we do not show these two AUI interfaces, instead describe their behavior. AUI cannot register analysts by itself, instead it will communicate with PrC to have an analyst registered and authenticated. The details of login/authentication have already been specified in Chapter 4. Below, we give only communication details between AnC and PrC.

- An analyst submits *username* and *password* information to AUI registration interface window.
- AUI sends it to PrC through interface API requesting the analyst to be registered.
- The registration process, as described in Chapter 4, is completed in PrC.
- PrC sends the *user_id* to AnC through interface PAI.
- The received *user_id* along with *username* and *password* are saved by AnC in the local variable *userSet*.

Registered analysts can sign in to construct queries through AUI. The communication details of login interface window in AUI is similar to the above.

5.2 Query Construction

Query construction is the primary task of AUI. Many researchers have recommended (Knight, 2021; Koesten et al., 2020; Mihaila et al., 2000; Strozyna et al., 2018) a user-centric and goal-oriented interface model to assist users construct rich queries, but no interface model that can construct rich queries exists. After some investigation, we found the *Goal Question Metric* (GQM) method (Caldiera & Rombach, 1994; Shull, Seaman, & Zelkowltz, 2006; R. V. Solingen & Berghout, 1999), the first method developed for software quality improvement and goal-oriented measurement, to be the most suitable model for integration into AUI. We are adapting this approach, especially the goal-oriented and query-centric parts, to collect information from analysts in putting together a rich query in which goal, semantics, and quality dimensions can all be combined. In order to enable analysts achieve their tasks without learning GQM method, we first designed the *Analysis Request Template* (ART) shown in Figure 5.1 in which the expert knowledge of GQM modeling is embedded. We use ART to drive the AUI interfaces for user dialogues during query construction.

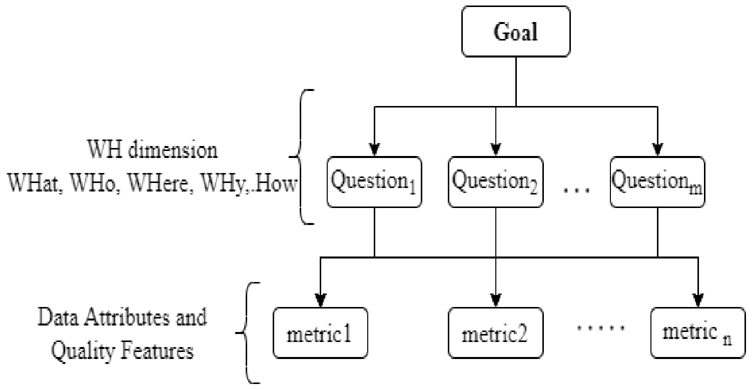


Figure 5.2: General View of GQM Model

5.2.1 Embedding Expert Knowledge of GQM in ART

GQM has three levels of goal definition and requirements collection. These levels, as shown in Figure 5.2, are *conceptual*,*operational*, and *quantitative*. So, we designed ART with three segments, each segment corresponding to each GQM level.

Conceptual Level: In the conceptual level, GQM method looks at the goal of the software to be developed, and helps to precisely define the project goal. According to (Shull et al., 2006; R. V. Solingen & Berghout, 1999), "GQM helps to *limiting data collection to what's needed to answer relevant questions, stating assumptions explicitly, and using a clear model for interpreting measurement results*". Thus, in the conceptual level GQM focuses on goal definition and breaks it down to the following elements:

(1) Object - What is being examined

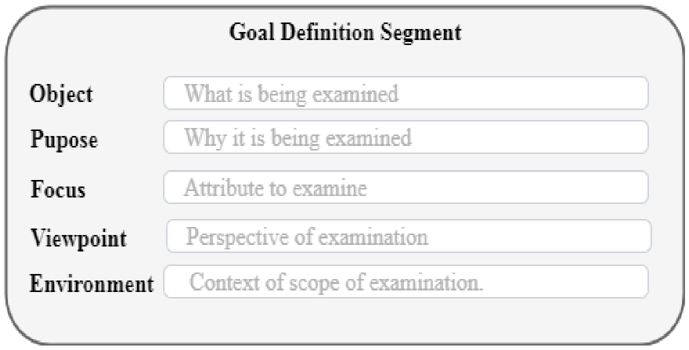
(2) Purpose - Why it is being examined

(3) Focus - Attribute to examine

(4) Viewpoint - Perspective of examination

(5) Environment - Context of scope of examination.

We adapt this to define the **Goal and Scope Segment** in ART, in which the goal and scope of intended analysis of data seekers are recorded. The GQM follows a "textual description" to construct the conceptual level details. For example, if the goal is to measure the accuracy of diabetes sensor, the textual description *"Analyze diabetes data produced by the sensor manufactured by X-company, to measure the accuracy level of diabetes with respect to quality feature from physician's perspective"* may be recorded in this segment. It identifies in the sentence, the syllabi "Analyze diabetes sensor data", "measure accuracy", "Quality feature", "Physician", and "X-Company" as representing respectively object, purpose, focus, viewpoint, environment. In ART we use a schematic approach. Figure5.3 shows the goal-specific information of the above example recorded in the "Goal and Scope Segment" of ART. This approach avoids full natural language processing. This goal definition template, explanation of the elements in it, and examples are part of ART segment and is stored locally in AnC as part of the *requestDirectory* variable. This segment provides the knowledge support for assisting the analysts while they construct queries through the interface window shown in Figure 5.9.


Goal Definition Segment

Object	What is being examined
Pupose	Why it is being examined
Focus	Attribute to examine
Viewpoint	Perspective of examination
Environment	Context of scope of examination.

Figure 5.3: Goal and Scope Segment of ART

Operational Level:‘ In the operational level of GQM, a set of questions is defined to determine the attributes needed for the goal assessment and achievement. Questions are constructed to help users refine and articulate the goal and represent it in attributes. For example, the question *How to measure sensor accuracy of diabetes patients?* moves the concept of measuring an accuracy to the operational level. The answer to the question can specify the attributes needed to measure accuracy. In addition, these answers can lead to attributes needed for evaluation to progress of tasks. That is, the defined attributes can lead analysts to full awareness of the progress related to goal assessment, achievement, and evaluation.

Inspired by the “questions concept” at the operational level in GQM, we designed the **Attributes Definition Segment** in ART. In it questions that are related to the context of current issue and purpose of the analysis task are recorded. That is, the template includes questions which focus on the goals/context elements defined in **Goal and Scope Segment**. In addition, this segment will include WH type questions shown in Figure 5.2. The sample questions provided in ART, listed below, could inspire analysts to put their own questions.

(1) What are the attributes used to describe the Object being examined?

(2) Which attributes are required to evaluate a Purpose being examined?

(3) Which attributes are needed for analysis?

(4) Which attributes “from the examination viewpoint” are required for this analysis goal?

(5) Which attributes are required to describe the examination environment?

(6) Is time frame essential for this analysis issues?

(7) Is there any specific geographical area specific for this analysis task?

(8) Any other attributes required for analysis goal achievement?

We included the last question to give the analysts some room to think of other attributes that could be needed for their goal. In many practical situations (Shull et al., 2006; R. V. Solingen & Berghout, 1999) the question-based GQM approach has benefitted the software project assessment. So, we believe that answering the stated questions in ART will help analysts define precisely goal and context-centric attributes with their metrics that are essential to achieve the analysis goal. The "attribute definition" interface window shown in Figure 5.7 is based on this ART knowledge segment.

Indeed, the set of questions in ART are limited to some extent. Yet, during query construction in Figure 5.9 the questions in ART might trigger the analyst to engage in a "two-way" search and match, between the attributes they are selecting based on ART queries and the tags (semantic concept terms) they can get from the LKC unit of PrC. That is, to answer each of the questions raised through ART interface in Figure 5.7, the analyst will be enabled by our design to browse through all the possible tags (concept terms) of the chosen domain/subdomain. That is, when the analyst clicks on the list to add the attributes in Figure 5.9, AnC sends to PrC the $(domain, subdomain)$ information of the query and requests the set of all the tags defined under this domain and subdomain in MTD. The analyst can browse through this list and discover that one or more of the tags are semantically close to the attribute of the query. All selected attributes are stored in AnC under *tagSet* variable. So, the knowledge received from PrC unit, put together with the "limited knowledge" discovered in answering ART questions might help the final selection of the attributes of the query under construction.

Quantitative Level: In GQM, precise answers that include specific modes for measurement of attributes are developed in the quantitative level. Precision is achieved through answers to metrics that can be associated with quality attributes. Two sample questions and answers in the operational level of GQM are given below.

(1) For the question *What medical devises are used for diabetes?*, answer may includes "Glucose meters", "Lancets", "Test Strips", and "Insulin pump".

(2) To add more details to measures questions like *How much data is needed?*, *Which format the collected data should be in?*, and *At what times the data should be collected* can be raised. The answers add quantitative and contextual details.

We adapt this approach to design **Quality Definition Segment** in ART. Instead of a query-based approach used in GQM, we use a "schematic" approach, assuming that the list of pre-defined quality dimensions for analysts, shown in Table 5.1, will be sufficient to answer the questions they may have on quality aspects to be included in their queries. Each

Table 5.1: Quality Dimensions Provided in ART Template

Quality Dimension	**Data Type**	**Value - Measurement**
Volume	Numerical	gigabyte
Variety	Enumerated Nominal	Nominal Values. e.g.{xml,cvs,text,...}
Velocity	Numerical	Number of days
Veracity	Enumerated Categorical	{not known, acceptable, verified}
Value	Numerical	Percentage
Release Year	Enumerated Numerical	{1995,....,2050}
Availability	Interval	[2021,2023]
Reliability	Enumerated Categorical	{low, medium, high}
Safety	Enumerated Categorical	{low, medium, high}

element in the table is associated with a type, unit of measurement, and description/example information. This table is included in ART. This segment of ART can be refined independently from other segments and re-evaluated whenever a new quality dimension needs to be added in future. The interface window in Figure 5.8 enables analysts to choose quality dimensions and associate weights and semantics during query construction.

The knowledge in ART is automatically invoked in the query construction steps to guide analysts to choose the right elements and measurements needed for constructing queries. This operational role of the three ART segments are explained in Section 5.2.2. The significant design features of ART are the following:

(1) ART is a "mini expert system" of GQM.

(2) An analyst need not know GQM methodology. ART guides an analyst with its expert knowledge.

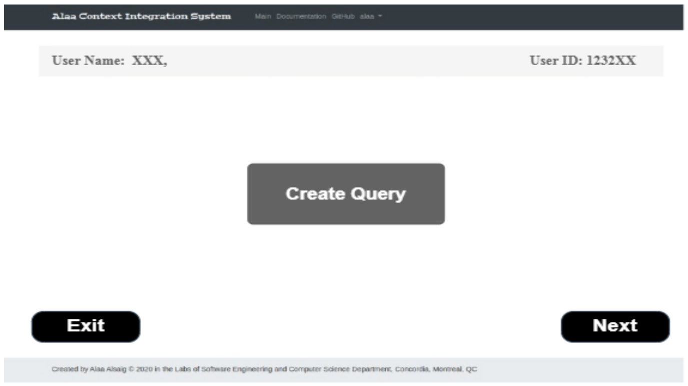
(3) ART is automatically enabled in the query construction steps in AUI interface windows.

(4) The role of **Attributes Definition Segment** is very significant. It provides expert help to collect attributes for the analyst query. These attributes are "user-centric". However, analysts will not use the attributes they select in the query. In AUI the "attribute interface window" in Figure 5.9, when invoked by an analyst, will contact PrC to get the set of semantic tags for the domain/subdomain pair and display it to the analyst. The analyst can choose the "best semantic tag" for each attribute selected with the help of ART and will use it in the query. In turn, the exact semantic

match of concept terms is made possible when the query is matched by CeC with the MTD in the PrC registry. So, the analyst will have the “actual attribute name” corresponding to “the name chosen by the analyst” after CeC send the AnC the list of ranked metadata descriptions. In turn, when the analyst downloads the set of datasets filtered by contextual contracts, the datasets will have in them the actual set of attributes. We explain more on this issue as part of future work section in Section 8.

5.2.2 Query Construction Steps in AUI

AUI is the front end of AnC provided to analysts to build their queries. Whenever an interface window in AUI requires data stored in PrC, AnC interacts with PrC through API interface to assist query construction. The elements domain, subdomain, attributes, quality dimensions, weights for attributes, and semantics for “best selection” are required to construct a query. In order to get information on these elements the analyst will use many interface windows of AUI. Many of these interface windows are similar to the PUI interface windows discussed in Chapter 4. So, we assume that the meanings of “interface buttons” are clear. Analyst interactions through AUI windows are described below.


Alaa Context Integration System
Main Documentation GitHub alaa
Create Query
Exit
Next
User Name: XXX,

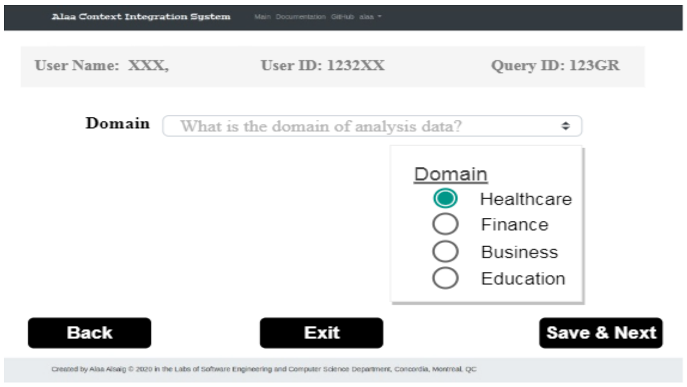
User ID: 1232XX

Created by Alaa Alsaig © 2020 in the Labs of Software Engineering and Computer Science Department, Concordia, Montreal, QC

Figure 5.4: AUI: Query Start Up Page

Query Startup: Every registered/authenticated analyst will be displayed with the “query start up” interface window shown in Figure 5.4. The analyst either exits or clicks on the “Create Query” to open the interface page in Figure 5.5 for requesting domain information. A new query identifier will be generated when “Create Query” button is clicked.

Select Domain: It is assumed the analysis is suitably motivated by ART **Goal Definition**


Alaa Context Integration System

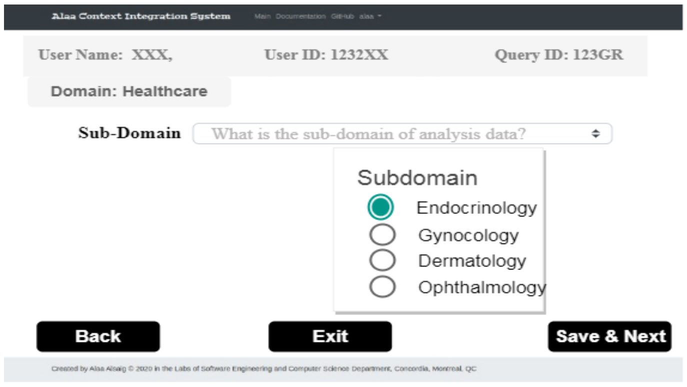
User Name: XXX, User ID: 1232XX Query ID: 123GR

Domain: What is the domain of analysis data?

Domain
[✓ Healthcare] Finance Business Education

Created by Alaa Alsaig © 2020 in the Labs of Software Engineering and Computer Science Department, Concordia, Montreal, QC

Figure 5.5: AUI: Domain Selection Interface


Alaa Context Integration System

User Name: XXX, User ID: 1232XX Query ID: 123GR

Domain: Healthcare

Sub-Domain: What is the sub-domain of analysis data?

Subdomain
[✓ Endocrinology] Gynocology Dermatology Ophthalmology

Created by Alaa Alsaig © 2020 in the Labs of Software Engineering and Computer Science Department, Concordia, Montreal, QC

Figure 5.6: AUI: SubDomain Selection Interface

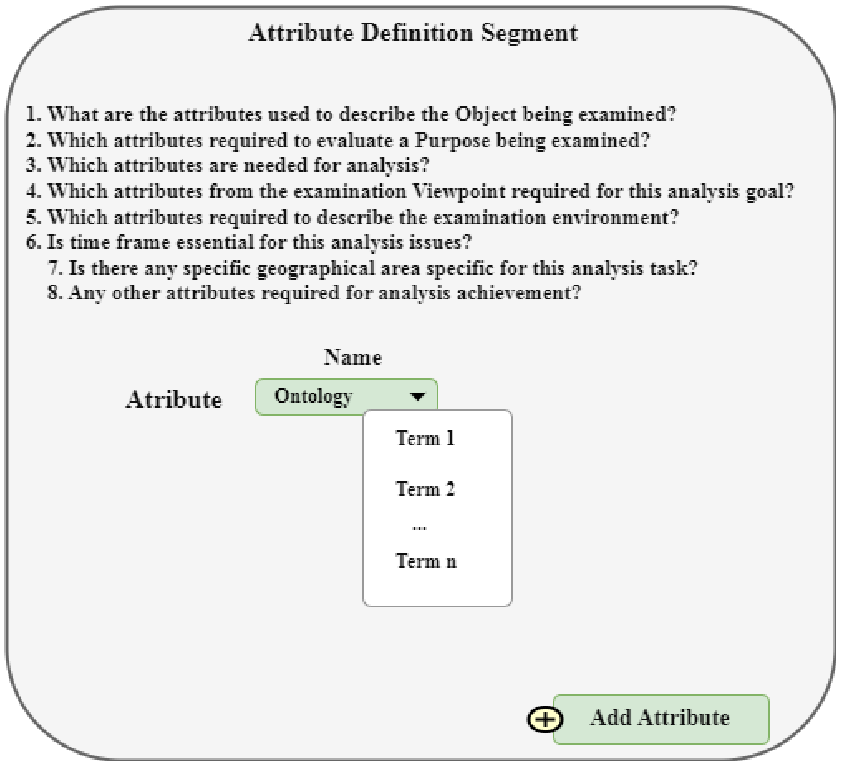
Segment to specific domain. By clicking the "text box" in the window shown in Figure 5.5, a scroll down panel with domain names will appear. The user can select one domain name, which will be entered in the text box. In order to display the set of domain names, AnC interacts with PrC through API to request for the set of domains it has. From its local storage the PrC copies the set of domains and sends to AnC through PAI. The received domain list is stored locally in AnC in *domSet* variable. This interaction is specified below.

Domain Request Interaction:

Message Sent by AnC to PrC: $domain\_request(user\_id)!$: The parameter is the "User Id" shown on this window.

Message Responded by PrC to AnC: $domain\_send(user\_id, D)?$: response sent by PrC. PrC sends a copy of set of domains from it's local storage, and is copied locally: $D = domSet$

The analyst clicks on "Next" button to go to the next step.


Attribute Definition Segment

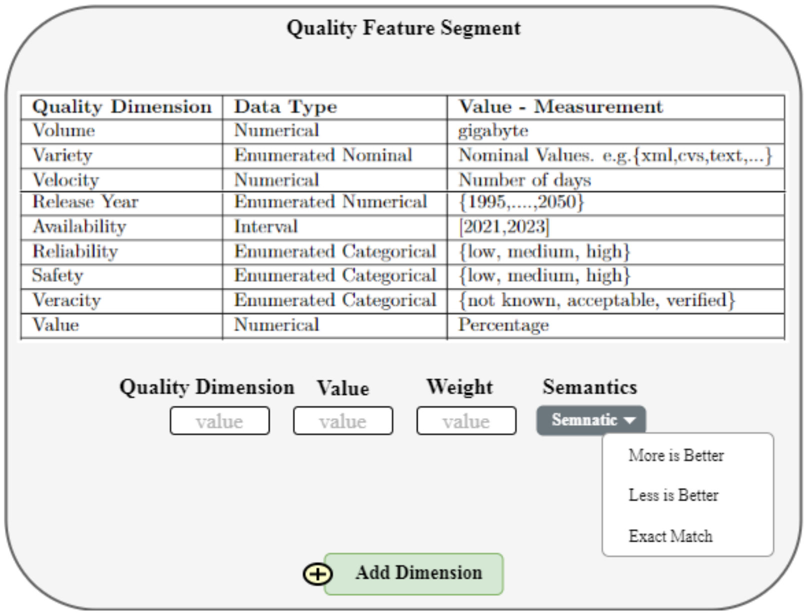
1. What are the attributes used to describe the Object being examined?
2. Which attributes required to evaluate a Purpose being examined?
3. Which attributes are needed for analysis?
4. Which attributes from the examination Viewpoint required for this analysis goal?
5. Which attributes required to describe the examination environment?
6. Is time frame essential for this analysis issues?
7. Is there any specific geographical area specific for this analysis task?
8. Any other attributes required for analysis achievement?

Figure 5.7: Attribute Definition Segment of Analyst Request Template (ART)

Select Subdomain: The selected domain name appears on the subdomain selection interface window shown in Figure 5.6. When the analyst clicks on the text box, a list of subdomains of the selected domain is provided to the analyst. In the background, clicking on the text box triggers AnC to interact with PrC and request the available subdomains of the domain selected by the analyst. In return, PrC sends the list of subdomains to the AnC. The received list is stored locally in AnC in a variable named *subdomSet*. The interaction specification being similar to the one for "domain selection" we skip the specification description. The analyst can click on the button "Back" if changing the domain is needed. If user clicks "Next" button the interface window in Figure 5.9 opens.

Query Construction using ART: The three parts in Figure 5.9 guides analysts to construct their queries, supported by the knowledge in ART.

- *First Part:* When the analyst clicks in the first text box in the "goal section" of this window, the ART page in Figure 5.3 opens with an example to assist the user to fill in this section. This information is only for analyst record. It will not be part of the constructed query structure.



Quality Dimension	Data Type	Value - Measurement
Volume	Numerical	gigabyte
Variety	Enumerated Nominal	Nominal Values. e.g.{xml,cvs,text,...}
Velocity	Numerical	Number of days
Release Year	Enumerated Numerical	{1995,....,2050}
Availability	Interval	[2021,2023]
Reliability	Enumerated Categorical	{low, medium, high}
Safety	Enumerated Categorical	{low, medium, high}
Veracity	Enumerated Categorical	{not known, acceptable, verified}
Value	Numerical	Percentage

Figure 5.8: Quality Features Specification Segment of ART

- Second Part: The domain and subdomain information are auto-filled. So, the user can click on the button "Get Knowledge" to fill the attributes section. The interface window in Figure 5.7 pops up. This interface, based on ART, lists the pre-defined questions and provides the interface button to invoke the ontology terms from PrC. To answer the questions and choose attributes, the analyst can click on the "ontology" button. AnC communicates with PrC as specified below:

  Tags Request:

  Message Sent by AnC to PrC: $tags\_request(user\_id, d, sd)!$: Parameters d and sd are respectively the domain name and sub-domain name on the interface page of Figure 5.9.

  Message Received by AnC from PrC: $tags\_send(user_id, tagsset)?$: AnC receives the set of tags $tagSet$ sent by PrC and saves it locally: $Tags(domain, subdomain) = tags$. The tags in $Tags(domain, subdomain)$ will be displayed to the analyst, who can select the appropriate tag for the attribute suggested through ART question.

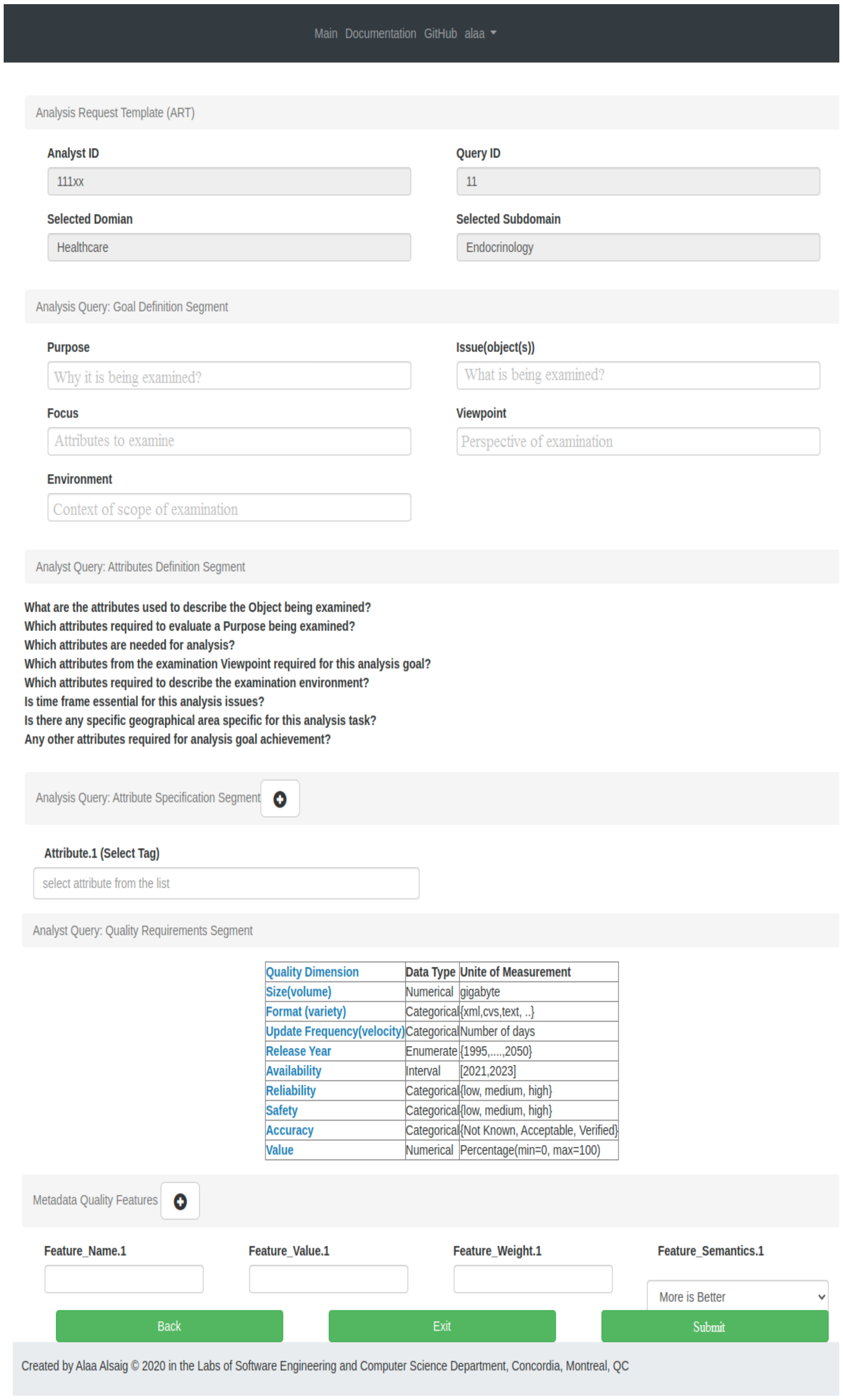
- *Third Part:* To complete the quality section part, the analyst clicks on the first text box. The interface window shown in Figure 5.8 pops up. This interface, backed by ART quality segment, guides the analyst to choose quality dimensions, units, weights, and semantics. In this interface, the analyst selects a dimension by clicking on the dimension name. This will fill the text box under the "Quality Dimension" part with the dimension name in Figure 5.9. For the selected dimension, the analyst can provide value and weight by clicking the corresponding buttons in the interface Figure 5.8. The value represents the preferred value of the analyst to this specific dimension. For example, if the analyst selected "Volume" as a quality dimension, then the value entered, say 120 GB, specifies the volume of data the analyst prefers in the dataset. Weight is a numerical value in the interval $[0, 1]$, which indicates the level of importance of this dimension to the analyst.

The 'Semantics" button can be enabled only if numerical type quality dimension is chosen. The numerical type quality dimensions are "Volume, Velocity, Veracity, Value, Release Year, Reliability, Safety". Notice that we have included "Veracity, Reliability, Safety" in this list because we have assigned the numerical equivalents to the categories. When one of these dimensions is chosen by the analyst, the interface enables the user to click on the button "Semantics" and choose one of the three displayed semantic options. We assume that an analyst understands the semantics of these options. The meaning of EM is that the analyst prefers the value specified in the query to be matched "as closely as possible" with the value of the selected quality dimension in metadata descriptions. The semantic option "More is Better" (MB) is interpreted by the system to look for "values in the metadata for the chosen dimension that are greater than the value specified in the query". The semantic option "Less is Better" (LB) is interpreted by the system to look for "values in the metadata for the chosen dimension that are less than the value specified in the query". For example, assume that the user query specifies the value 120 for Volume dimension GB, and the value for Volume in one metadata md_1 is 110 GB and the value for Volume in metadata md_2 is 130 GB. If MB semantics is selected by the user for Volume dimension, the similarity score algorithm in Chapter 6 will assign a higher score to md_2 than md_1. Selected quality dimensions and their values, weights, and semantics are all stored locally in AnC respectively in $qualityRecord[dimension : value]$, $weightRecord$, and $semanticRecord$.

Once the three parts in Figure 5.9 are completed, the analyst may click on the button “Submit” to save the constructed query in the form shown in Figure 5.10. This *query_vector* includes the collected information and their types. In case changes are required, the analyst can click on the “Back” button to return to the previous window shown in Figure 5.6. In addition, the analyst can end the session without any operation to perform by clicking on “Exit” button. Below we give an example query. For the sake of convenience we represent the three semantics as numerical values, assigning 1 for MB (More is Better), 0 for EM (Exact Match), and -1 for “Less is Better”.

Listing 5.1: Query Vector Example

```
{request_id: Req_1,
user_id:analyst_1
Domain: "Healthcare",
Subdomain: "Endocrinology",
attributes:["Patient’s Age","Gender", "Diabetes Type", "Blood_Pressure", "Hba1c"],
quality_feature:{(size, 50), (format,"xml,cvs"), (release_year,2021), (value,
    0.8)}
weights:[1,0.5,0.3,0.9],
semantics:[1,0,1,1]
}
```



Analysis Request Template (ART)

Analyst ID: 111xx

Query ID: 11

Selected Domian: Healthcare

Selected Subdomain: Endocrinology

Analysis Query: Goal Definition Segment

Purpose: Why it is being examined?

Issue(object(s)): What is being examined?

Focus: Attributes to examine

Viewpoint: Perspective of examination

Environment: Context of scope of examination

Analyst Query: Attributes Definition Segment

What are the attributes used to describe the Object being examined?
Which attributes required to evaluate a Purpose being examined?
Which attributes are needed for analysis?
Which attributes from the examination Viewpoint required for this analysis goal?
Which attributes required to describe the examination environment?
Is time frame essential for this analysis issues?
Is there any specific geographical area specific for this analysis task?
Any other attributes required for analysis goal achievement?

Analysis Query: Attribute Specification Segment

Attribute.1 (Select Tag): select attribute from the list

Analyst Query: Quality Requirements Segment

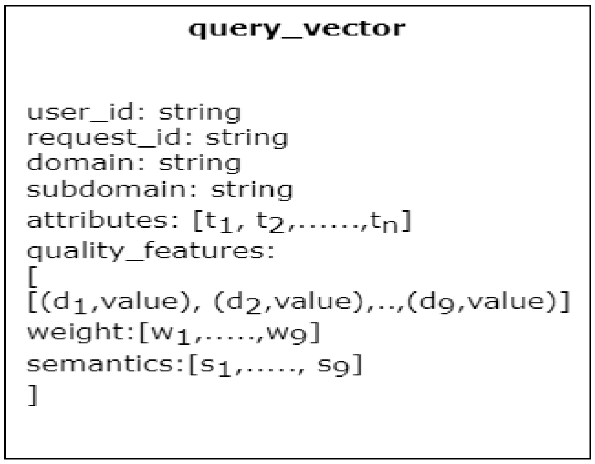
Quality Dimension	Data Type	Unite of Measurement
Size(volume)	Numerical	gigabyte
Format (variety)	Categorical	{xml,cvs,text, ..}
Update Frequency(velocity)	Categorical	Number of days
Release Year	Enumerate	{1995,....,2050}
Availability	Interval	[2021,2023]
Reliability	Categorical	{low, medium, high}
Safety	Categorical	{low, medium, high}
Accuracy	Categorical	{Not Known, Acceptable, Verified}
Value	Numerical	Percentage(min=0, max=100)

Metadata Quality Features

Feature_Name.1	Feature_Value.1	Feature_Weight.1	Feature_Semantics.1
			More is Better

Created by Alaa Alsaig © 2020 in the Labs of Software Engineering and Computer Science Department, Concordia, Montreal, QC

Figure 5.9: ART-assisted Query Construction Interface



query_vector

```
user_id: string
request_id: string
domain: string
subdomain: string
attributes: [t1, t2,......,tn]
quality_features:
[
[(d1,value), (d2,value),..,(d9,value)]
weight:[w1,.....,w9]
semantics:[s1,....., s9]
]
```

Figure 5.10: Generated Query Vector from AUI interface

5.3 Query Submission and Receiving Ranked Metadata

A completed query, stored locally in an array QV, is sent to CeC through the interface ACI with a request to receive the ranked list of metadata which the query features. In Chapter 6 we explain the algorithmic details which CeC will use to respond to this request. The ranked metadata list will be sent by CeC through the interface CAI. It will be received at the interface ACI of AnC, and is stored locally in $metadataList$. The interface specifications for these communications are traightforward and we skip the details.

5.4 Filter Metadata Using Contextual Contract

The analyst who receives the ranked list of metadata executes the following steps to filter a ranked sub-list of metadata that satisfy the local contexts of the analyst. This validation process may not be simple for all rules. In fact, defining formally "checking contextual compatibility" and automatically applying the formal definition to validate context compatibility are in general hard "open problems." As such it is not part of our prototype implementation. In Chapter 8 we propose formal definitions of context compatibility based on the works (Akman & Surav, 1996; Alsaig, 2017, 2022; Alsaig, Alagar, & Shiri, 2019; Wan, 2006), and suggest our future work on automatically validation of contextual compatibility. At present, we are assuming that the analyst will resolve contextual part only manually, using an approach similar to Delphi (Strozyna et al., 2018). **Delphi Method:** For each metadata x in $metadataList$ received from CeC, the analyst will manually execute the following steps with help from a group of experts.

(1) Let $y = Contract(x)$. That is, y denotes the contract part of the metadata description x.

(2) Validate (manually) each legal rule in $LegalRule(y)$. For example, if a rule states "data may not be shared with third parties" the validation is either to "accept" or "reject". By communicating this decision to the service provider through the "URL" or "Contact" information given in the metadata description, the analyst and the provider of the dataset agree on whether or nor dataset will be delivered.

(3) For validating SPC and DSC contexts, the analyst with experts may have to consult a

Trusted Authority to get to know whether or not the service provider is “trustworthy” and the dataset qualities have been certified. Based on that information and the recommendation of experts in Delphi group, the analyst will either accept or reject the dataset provider.

(4) For validating SDC contexts, the analyst can manually check whether the local context at service delivery location fulfill the requirements stated for each dimension in each SDC. For example, the privacy/security/legal aspects in the region where the analyst requires the dataset delivery must be acceptable to the contextual information in SDC.

(5) If the above validations are successfully completed, then the analyst can proceed to get the dataset of the metadata x.

Chapter 6

Detailed Design of CeC

CeC is the central piece that provides the tight coupling in TCF framework. It is the "dataset search engine", that receives a query from an analyst and delivers a ranked list of metadata descriptions that best match the query. So, the core functionalities of the engine are matching and ranking, which are achieved respectively by *Matching Unit* (MU) and *Ranking Unit* (RU) shown in the CeC design in Figure6.1. In Section 6.1 we give an overview of this design. We explain in Section 6.2 how CeC manages the functionalities through interface communications of with AnC and PrC. The matching method, based on *Formal Concept Analysis (FCA)* method is explained in Section 6.3.1. Ranking, based on a formal multi-feature similarity calculation, is elaborated in Section 6.4. This chapter is concluded in Section 6.4.3 with an example.

6.1 An Overview of CeC Design

CeC receives a query from AnC through its interface CAI and saves it internally in $queryVector$. The part of the query that contains domain (d), sub-domain (sd), and attribute set ($attSet$) information in the $queryVector$ will be used by MU and the "quality-features" (QF) part of the query will be used by RU. CeC sends (d, sd) pair to PrC through the interface CPI and requests for the "set of all pairs" $(mdid, tagSet)$, where $mdid$ is the identifier of a metadata description published in the PrC metadata directory for (d, sd), and $tagSet$ is the set of all tags in the metadata identified by $mdid$. CeC saves it in MTD variable locally defined in MU and performs matching operation using $attSet$ part

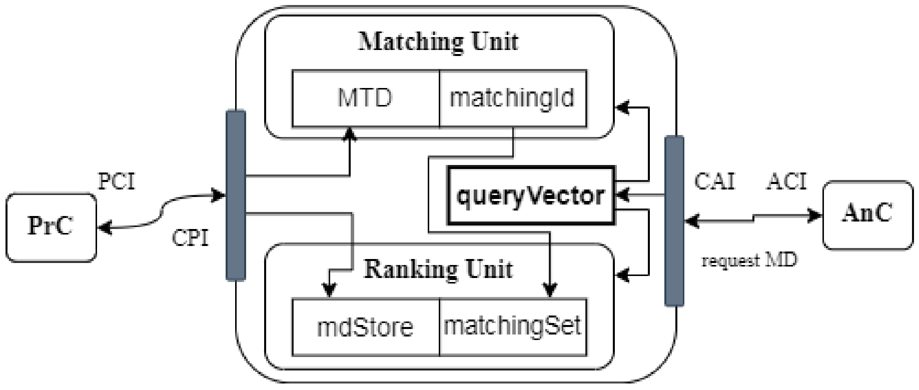


Figure 6.1: Abstract View of Coupling Engine Component (CeC)

in the query. The set of matched metadata identifiers (*mdid*s) are saved by MU in the variable $matchingId$. Then, it calculates "Value" for every metadata whose identifier $mdid \in matchingId$ using the $tagSet$ of this $mdid$ and the $attSet$ in the $queryVector$. The details are discussed in Section 6.3.1. Every $mdid \in matchingId$ is now replaced with the pair $(mdid, value)$ in $matchingId$. This set is copied into the variable $matchingSet$ of RU

RU requests for and receives from PrC the full metadata description for every $mdid$ in $matchingId$, and saves them in the variable $mdStore$. For each element $(mdid, value) \in matchingSet$, it chooses the metadata description with identifier $mdid$ from $mdSet$, and fills in the "Value" field of the metadata description with "value". Next, it starts preparing the quality dimensions of every metadata from $mdSet$ in order to perform similarity calculation with the "quality-features" (QF) part of the query in $queryVector$. In Section 6.4 we explain the query structure preparation, "scoring function definitions" for quality attribute types, similarity computation based on them, and ranking.

6.2 Interfaces of CeC Component

CeC communicates with PrC component through interface CPI, and communicate with AnC component through interface CAI. The functionalities of CeC depend on the input coming from both interfaces. Hence, before discussing CeC functionalities, the interfaces of CeC are explained in the following sections.

6.2.1 Interface Communications between AnC and CeC

CeC receives from AnC the message $metadata\_request(user\_id, QV)$, where $user\_id$ is the analyst's registered identification in the system, and QV is the query constructed by the user. The structure of QV has been explained in Chapter 5. Because only registered analysts can construct queries, CeC does not check the validity of user. It saves QV in the local variable $queryVector$. After ranking is completed, CeC sends the message $metadata\_send(user\_id, ranked\_metadataSet)$, where $ranked\_metadataSet$ is the ranked list of metadata descriptions, to the user (at AnC) through its interface CAI. This communication being simple, no formal description of it is given.

6.2.2 Interface Communication between CeC and PrC

The first communication between CeC and PrC is initiated when CeC sends to PrC a message with parameters *domain* and *subdomain* through the interface CPI, requesting the set of pairs (the identifier of metadata, the set of tags in that metadata description) stored in its directory corresponding to the pair $(domain, subdomain)$. In response, PrC sends the requested set of pairs. The second communication is repeated many times, once for every $mdid \in matchingId$, wherein CeC will request the full metadata description from PrC. Below we explain these communications and how the components respond. We show only message names, and the internal actions triggered by them.

First Communication:

Message sent by CeC: $tagsetPair\_request(d, sd)!$.

The parameters d and sd are copied from $queryVector$. So, all parameters are valid. This message, received by PrC at the interface PCI, is validated. That is, $d \in domSet \wedge sd \in subdomSet$ must be true for PrC to respond.

Response sent by PrC: $tags\_send(d, sd, tagsetPair)?$.

Here $tagsetPair$ is the set $\{(mdid(x), tagSet(x)\}$ of pairs, where

- $x \in MI[d, sd]$; that is, x is a metadata in the directory $MI[d, sd]$;
- $tagSet(x)$ is the set of tags in the "Attribute" part of the metadata model x. Abstractly, this set is calculated as $Tags(Attribute(Functional(Service(x))))$.

This message, when received at CeC through the interface CPI, the set $\{(mdid(x), tagSet(x)\}$ of pairs is copied into the local variable MTD.

Second Communication:

Message sent by CeC: $metadata\_request(mdid)!$. The parameter $mdid$ is extracted from the pair $(mdid, value) \in matchingSet$. So, it is a valid identifier of a metadata in PrC directory.

Response sent by PrC: $metadata\_request(mdid, metadata)!$. Here $metadata$ is the full metadata description from the directory of PrC and its identifier is $mdid$. Because every metadata has a unique identifier the property $x = metadata \wedge x \in MI[d, sd] \wedge mdid(x)$ is true. When this message is received by CeC in its CPI interface, the parameter pairs $(mdid.metadata)$ will be saved in its local variable $mdStore$.

6.3 Functionalities of Matching Unit (MU)

The following are the two functionalities of MU.

- *Semantic Matching*: For every element $x = (a, b)$ in the set MTD, select a if b semantically matches the $attSet$ part in $queryVector$.
- *Measure Semantic Closeness*: Using the sets b and $attSet$ determine $value(a)$, which is a measure of semantic closeness of the metadata with identifier a to the analyst stated set of attributes. We use Jaccard measure (P. Zhang, Wang, Hu, & Sorrentino, 2014), a set-theoretic measurement of closeness between feature sets, to measure $value(a)$.
- *Communicate with RU*: Send the set of pairs $(a, value(a))$, for every a selected in the first step.

For the purpose of our current modeling of metadata, we can solve the matching problem in a straightforward manner, using "inverted lists", or using "0-1 matrix representation, or just selection through exact match between tags in the query and tags extracted from metadata descriptions. However, we decided to use basic fundamental principles of *Formal Concept Analysis* (FCA) (Alsaig, Mohammad, & Alsaig, 2016; Cimiano, Hotho, & Staab, 2005; Poelmans, Ignatov, Kuznetsov, & Dedene, 2013) for semantic matching, motivated by the following two reasons.

- FCA is a formal method to define *relations between objects and their properties.* The two formal concepts *Extent* and *Intent* define the extraction of "the set of properties related to a given object" as well as the extraction of "the set of objects related to a property". FCA gives a *lattice structure* to organize object-property pairs to explain all possible partial order structures in a relation involving a large number of objects and their properties. These fundamental definitions are sufficient to solve our matching problem. The set extracted using these definitions are both *correct* and *complete.* That is, no separate proof is necessary to argue that "nothing is left out". The second advantage is, it can deal with "very large relations", which we expect to be the norm with Big Data.

- In our current model the tags are the "leaf nodes" of the ontology in OnC component that supports PrC and AnC activities. In future, we can relax this restriction, and allow the concept terms used to describe metadata attributes to belong to any level in the ontology. In this extended model, "exact match" cannot be used. We must use ontology-based semantic relation between tags and replace "exact much" by " semantic relatedness" in FCA concept definition. The paper (Poelmans et al., 2013) discusses FCA-based analysis of knowledge-based systems and these methods can be used by MU in the matching phase. In Chapter 8 we explain more on other scenarios where tags will have richer semantics and how FCA tools should necessarily be brought into MU.

6.3.1 Formal Concept Analysis (FCA)

FCA is a mathematical theory of concept hierarchies based on Lattice Theory (Cimiano et al., 2005). FCA can be applied to many application domains as long as the objects of interest in the domain and their attributes (properties) are formally defined in a relation. In the thesis we use only basic concepts of FCA for describing datasets through their attributes and achieve semantic matching between query and datasets.

In FCA the "formal context" notion is to define the "scope of the objects and attributes" for a specific domain. This context notion is not to be confused with contexts that we use for metadata descriptions and dataset delivery in this thesis.

Definition 6.3.1. *(FCA Formal Context):*
A triple (G, M, I) is a formal context if G and M are non-empty sets and $I \subseteq G \times M$ is a binary relation.

In an application, the elements of G are objects of interest and the elements M are attributes that sufficiently will describe all objects, and $G \cap M = \emptyset$. So, relation I has pairs $< g, m >$, $g \in G$, $m \in M >$ that are related (by the underlying semantics of I).

We consider only non-empty subsets of G and M. For every subset $A \subseteq G$, the set $A' := \{m \in M | \forall g \in A : (g, m) \in I\}$ defines the set of attributes for objects in A. For every non-empty subset $B \subseteq M$ of attributes, the set $B' := \{g \in G | \forall m \in B : (g, m) \in I\}$ defines the set of objects that have the attributes B. The sets A' and B' are *maximal* in the sense that "it is complete" in extracting the respective elements.

Definition 6.3.2. *(Formal Concept):*
A pair (A, B) where A is a set of formal objects and B is set of formal attributes that are closed *(i.e; one can neither enlarge the attribute set nor the object set) within a defined context is called a formal concept of the context (G, M, I) if and only if $A \subseteq G$, $B \subseteq M$, $A' = B$ and $A = B'$. The set A is called the* extent *of the formal concept (A, B) and the set B is called its intent. The sets A and B are maximal.*

Observing that in data mining, software engineering, and big data analysis it becomes necessary to deal with higher-order concepts to process clusters, concepts, and their associations several generalizations of FCA definitions have been proposed by the authors (Kwuida, Missaoui, Balamane, & Vaillancourt, 2014; Valtchev, Hacene, & Missaoui, 2003; Valtchev, Missaoui, & Lebrun, 2002). They discuss a variety of algorithms on the generalized formal concepts. The most fundamental generalizations among these are $exists$, $forall$, and α which we explain below. In Chapter 8 we comment on how these generalizations and rest of their work can be used in our future work on generalizing TCF.

Definition 6.3.3. *(Concept Generalization):*
($\exists$) $gJs :\Leftrightarrow \exists m \in s,\ gIm$. Consider an information table describing employees from different cities. We start to initialize the context whose objects are "employees" and whose attributes are the "cities" where these employees live in. If there are too many employees, we can group them into "provinces" where they live to reduce the number of attributes. Then, the

new attribute is now a "province" P whose elements are "provinces". It is natural to state now that an employee is from province p if the employee is from a city g, where g is a city that is located in province p. Formal concepts under this generalization can refer to "cities in a province" as a sub-relation.

Definition 6.3.4. *($\forall$) $gJs :\Leftrightarrow \forall m \in s, gIm$. Consider an information system about diseases, patients, and a set of symptoms. Assume that components are: $symptom_1$, $symptom_2$, and $symptom_3$, and a patient is considered "diagnosed with disease A" if the patient has all symptoms. The objects of the context are Patients and the attributes are the different symptoms the patient might have. Then, if we group together the different attributes, it summarizes the patients diagnosed with disease_A for example*

$$symptom_1,\ symptom_2,\ and\ symptom_3 \rightarrow disease\_A$$

Definition 6.3.5. *(α) $gJs :\Leftrightarrow \frac{|\{m \in s | gIm\}|}{|s|} \geqslant \alpha_s$, where α_s is a threshold that is defined by the user for the attribute s. This case can be used to generalize ($\exists$) case with $\alpha = \frac{1}{|M|}$ and ($\forall$) case with $\alpha = 1$.*

In the thesis, the FCA concepts $extent(y)$ and $intent(y)$ are sufficient for matching process because domain knowledge is the set of leaf nodes which are atomic and we are dealing with one ontology of a finite size. So, if $y = (A, B)$ is a formal concept under a defined formal context, we use operations $extent(y)$ and $intent(y)$ to extract respectively the sets A and B. However, the extended cases ($\exists, \forall,$ *and* α) are essential to manage huge number of concepts and properties coming from different levels of an ontology as well as from multiple ontologies.

6.3.2 FCA-based Matching

We map the problem we have to a problem in FCA and solve the matching problem using basic FCA definitions. We treat the metadata identifiers in MTD as objects, and the set of all tags in the union of *tagSets* in MTD as attributes of objects. Formally, in FCA terminology we write

$$G = \{mdid | mdid = First(x), x \in MTD\}; \qquad M = \bigcup_{x \in MTD} Second(x)$$

We define the relation $I = has$ on $G \times M$. That is, $I \subseteq G \times M$, $(g, m) \in I$ means that metadata $g \in G$ "has" the tag $m \in M$. Switching to the notation in our model, the pair $(mdid_i, tag_j)$ is a member of I if the metadata with identifier $mdid_i$ has the tag tag_j. Thus we have defined the formal context (G, M, I) for our problem, in which many formal concepts exist. In particular, each element $x \in MTD$ is a formal concept if we rewrite the first element of x as a singleton set. That is, if $x = (mdid, tagSet$, then we rewrite it as $FC(x) = (\{mdid\}, tagSet)$. We form the union of these concepts

$$MTD_{FC} = \bigcup_{x \in MTD} FC(x)$$

The set MTD_{FC} is the set of all formal concepts for our matching problem. The *attSet* in the query is a set of attributes. However, it is possible that $attSet \nsubseteq M$, and therefore a concept with *attSet* as "intent" may not exist in the set MTD_{FC}. So, we have to search the set of all formal concepts in MTD_{FC} and select every extent of a concept in it for which its corresponding intent has a non-empty intersection with *attSet*. Formally, for $y \in MTD_{FC}$, define $result(y)$ as

$$result(y) = extent(y), \quad \text{if} \quad intent(y) \cap attset \neq \emptyset$$

Therefore, $result(y)$ is a singleton set $\{mdid\}$ that corresponds to $intent(y)$. We need to take the "union" of all $result(y)$, for every $y \in MTC_{FC}$. Hence, the result (*result*) of matching is the set of all metadata identifiers in *result*, defined below:

$$result \ = \bigcup_{y \in MTD_{FC}} result(y)$$

By FCA definition of formal concept, the "intent" and "extent" of concepts are "maximal". Hence, the value of *result* is "maximal". That is, it has "all metadata identifiers" of matching procedure, and nothing that matches user query has been left out.

6.3.3 Calculating Value

For every $mdid \in MTD_{FC}$ we calculate a value, based on the "semantic closeness" of the *attSet* to the *tagSet* associated with *mdid*. We adapt Jaccard measure (Harispe, Sánchez, Ranwez, Janaqi, & Montmain, 2014; P. Zhang et al., 2014), which defines for sets A and B the similarity measure

$$JS(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

We define "the ratio of the amount of overlap between the set of attributes in the query with the set of tags in the metadata model to the number of attributes requested in the query" as the "semantic closeness measure" of a $tagSet$ to $attSet$. So, we redefine the above formula as below for calculating the "value" of the metadata whose identifier $mdid \in\ result$

$$value(mdid) = \frac{|tagSet \cap attSet|}{|attSet|} \tag{1}$$

The set of pairs $\{mdid, value(mdid)\}$, $mdid \in result$, is saved in $matchingId$ variable in MU, and is sent to RU which saves it in $matchingSet$ for ranking purposes.

6.4 Ranking Metadata

RU requests PrC for the full metadata description for every $mdid \in MTD_{FC}$. The received metadata descriptions are saved in $mdStore$. The ranking algorithm is applied to the elements in $mdStore$. The metadata descriptions in $mdStore$ are ordered according to non-increasing "similarity" measure that will be computed by comparing the quality features in $queryVector$ with the quality features in every metadata model saved in $mdStore$. The ranking algorithm used in this work is influenced by the *Fairness Perspective*, both from algorithmic and user-centric perspectives. discussed in the work (Alsaig, Alagar, Mohammad, & Alhalabi, 2017; Bawazir, Alhalabi, Mohamed, Sarirete, & Alsaig, 2018). From the user perspective we have the following features:

(1) Many types of quality values can be included in a query, as explained in Chapter 5. The types allowed are "numerical" type, "ranked categorical type" (with numerical equivalents), "nominal type", and "interval type". Our design can be extended with other user defined types.

(2) The query allows analysts "to specify semantics in order to look for better metadata". They can specify a weight for each attribute to indicate the level of significance.

Semantics for an attribute may include mode of comparison and an indication of "what is better?". The two modes for comparison are *Exact Match* (EM), and *Best Match* (BM). The BM mode is allowed only for numerical type values and ranked categorical values. In this mode the analyst can specify either *More is Better* (MB) or *Less is Better* (LB). When no semantics is specified the default mode is set to EM. In Section 6.4.1 we explain the semantics in more detail.

Design features from algorithmic perspective are the following:

- We recast the quality dimensions in a specific order, and get a standard vector-based model for quality dimensions. If n is the total number of quality dimensions (in our case, $n = 9$, both the query vector qu and the query vector qm for quality dimensions in a metadata have (1) the same order of quality attributes, and (2) have length n.

- Not all quality dimensions may be chosen by either the service provider or the analyst. However, we want all the components of qu and qm have values. This we achieve by filling the "unspecified" quality attributes in the vector with 0s. Hence, the quality vectors for ranking will have the same order of attributes and have length 9.

- Qualities have different types and hence similarity assessment cannot be done on the "whole quality vector". Consequently, we ignore many of the commonly used "distance-based" functions, and set-theoretic functions (Harispe et al., 2014). In order to effectively apply user semantics and weights, which are defined at attribute level, similarity assessment is best done at the level of attributes. Each quality attribute will require operations specific to it. So, we introduce similarity assessment function, called "score", to compute the similarity between every attribute pair in the query and the metadata.

- We want formality, simplicity, accuracy, and precision in score functions, in order to fairly compare and evaluate results.

We explain quality vector structure and give an overview of our algorithm in Section 6.4.1. The functions that we use to calculate the "scores" that measure similarity at attribute levels are discussed in Section 6.4.2. An example to illustrate our ranking method is given in Section 6.4.3.

6.4.1 Quality Vector Structure and Algorithm Overview

The ordering of quality dimensions listed in Table 5.1 of Chapter 5 is the order of attributes (dimensions) of every query vector. That is, Volume is the first attribute, Variety is the second attribute, and so on. Let qu denote the quality vector for the quality values specified in an analyst query. In this query $qu[i]$ is set to 0 if the i^{th} quality attribute is not given a value in the query, otherwise it's value is the value specified in the query. Similarly, let qm denote the quality vector for the quality values specified in a metadata description. In this query $qm[i]$ is set to 0 if the i^{th} quality attribute is not given a value in the metadata description, otherwise it's value is the value specified in the metadata description.

The algorithm compares values in $qu[i]$ and $qm[i]$, for $i = 1, \cdots, 9$. Depending upon the type, mode, and semantics for this attribute pair, it selects the appropriate scoring function from the list given in Section 6.4.2 and calculates $score_i$. Then, it uses the specified weights w_i, $i = 1, \cdots, 9$ to calculate the similarity between qu and qm as the weighted sum

$$s_{qm} = sim(qu, qm) = w_1 \times score_1 +, \cdots, w_9 \times score_9$$

Having calculated s_{qm}, for every qm constructed from the quality dimensions specified for metadata in *mdStore*, the algorithm sorts the metadata in *mdStore* in non-increasing order of the measures s_{qm}. The ranked list is sent to AnC through the interface CAI.

6.4.2 Scoring Functions

In this section we define scoring functions for different quality attribute types. In the list of quality attributes given in Chapter 4, we have three types of attributes. Attributes "Volume, Velocity, Value, Release Year" are of type Numeric". The ranked categorical type attributes are "Veracity, Reliability, Safety" and these are assigned "numerical equivalents" to their respective category ranks. So, we use scoring functions defined for numerical type to calculate scoring functions for these attributes. The attribute "Availability" takes an "Interval Value". We define a scoring function to compare "Intervals" and assess the score for two given intervals.

In (Alsaig, 2013) an extensive list of similarity functions are given and are compared.

Based on it, the basic function that we have chosen for "numerical type" is called "Relative Change", originally defined for vectors with numerical components. For two vectors $\{a_1, a_2, \cdots, a_n\}$ and $\{b_1, b_2, \cdots, b_n\}$, the relative change similarity measure is defined as

$$S_{RC} = \sum_{i=1}^{n} \frac{|a_i - b_i|}{max(a_i, b_i)}$$

This function is simple and precisely calculates "how much the values deviate". We adapt this function to define the "score" ("closeness" between two attribute) of similarity between the values of numeric type. If r and q are respectively two values, we define the score of similarity between them as

$$1 - \frac{|r - q|}{max(r, q)}$$

We use variations of this function to calculate scores under different modes of comparison and semantics. Below, we use the notation r and q to respectively denote the values in metadata and query for the quality attribute under discussion.

Scoring Function for Nominal/Categorical Attribute Pair

When ontology support exists for semantics of terms that are of type nominal or categorical, only "exact match" (EM) is possible. So, for two nominal values r and q the scoring function is defined as in Equation 2. For the quality attribute "Variety" the following function will be applied.

$$score(r, q) = \begin{cases} 1 & r = q \\ 0 & r \neq q \end{cases} \tag{2}$$

Scoring Function for Numeric Type and Ranked Categorical Type Attribute Pair

For ranked categorical type attributes in our study, we have assigned numerical equivalents as follows:

- *Attributes "Reliability" and "Safety"*:

low = 1; medium = 3; high = 5

- *Attribute "Veracity"*:

$$\text{not known} = 0; \qquad \text{acceptable} = 3; \qquad \text{verified} = 6$$

With this numerical values, we can use the scoring functions defined for numerical type. Hence, for the quality attributes "“Volume, Velocity, Value, Release Year, Veracity, Reliability, and Safety” the functions defined below will be used. We consider the options EM, BM with LB semantics, and BM with MB semantics.

- *Exact Mode (EM):* The similarity score is 1 only if the values match. Otherwise, instead of assigning 0 we assign "the relative change measure" to the score.

$$score(r,q) = \begin{cases} 1 & r = q \\ 1 - \frac{|r-q|}{max(r,q)} & r \neq q \end{cases} \tag{3}$$

  This function is symmetric, and normalized to have values in $[0, 1]$.

- *Best Match (BM) with MB Semantics:* We want to "reward" the cases where $r > q$ and "penalize" the cases where $r < q$. So, we consider the two cases:

  - case $r > q$ We have $max(r,q) = r$ and want $score(r,q)$ to be greater than 1. We achieve this when we add the "relative change" $\frac{|r-q|}{max(r,q)} = \frac{|r-q|}{r}$ to 1, and assign it to $score(r,q)$.
  - case $r < q$ We have $max(r,q) = q$ and want $score(r,q)$ to be less than 1. We achieve this when we subtract the "relative change" $\frac{|r-q|}{max(r,q)} = \frac{|r-q|}{q}$ from 1, and assign it to $score(r,q)$.

  Thus, the score function is as defined in Equation 4.

$$score(r,q) = \begin{cases} 1 & r = q \\ |1 - \frac{|r-q|}{q}|, & r < q \\ |1 + \frac{|r-q|}{r}|, & r > q \end{cases} \tag{4}$$

- *Best Match (BM) with LB Semantics:* We want to "reward" the cases where $r < q$ and "penalize" the cases where $r > q$. Hence, we just reverse *score* functions defined for MB semantics. The scoring function for LB semantics is as shown in Equation 5.

$$score(r,q) = \begin{cases} 1 & r = q \\ |1 + \frac{|r-q|}{q}|, & r < q \\ |1 - \frac{|r-q|}{r}|, & r > q \end{cases} \tag{5}$$

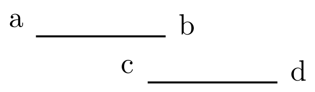
Scoring Function for Interval Type Attribute Pair

The quality attribute "Availability" has interval type because the underlying semantics of availability is *duration* measured as the number of years. If the interval $[2021, 2025]$ is assigned as value for availability, the meaning that we give to is "the strat dtate of availability is the beginning of year 2021, and the termination of availability is the last day of year 2022.

Let $[a, b]$ denote the interval in a metadata and $[c, d$ denote the interval in a query. If the two intervals are identical or if the interval $[c, d]$ is contained in the interval $[a, b]$, then the scoring function for this pair of values returns the value 1. If the two intervals do not overlap, the scoring function returns 0 for this pair. For other configurations of intervals, the scoring functions are calculated based on the "proportion of match (overlap) to the entire expectation". These are defined below:

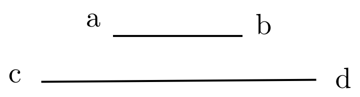
(1) $a <= c < b < d$. That is, the metadata interval $[a, b]$ partially overlaps on the left with the query interval $[c, d]$. The scoring function that measures the proportion of coverage is

$$\frac{b - c + 1}{d - c + 1}$$



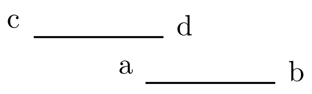
(2) $c < a < b < d$. The metadata interval $[a, b]$ is entirely within the query interval $[c, d]$. The scoring function that measures the proportion of coverage is

$$\frac{b - a + 1}{d - c + 1}$$



(3) $c < a < b <= d$. The query interval partially overlaps on the left with the metadata interval $[c, d]$. The scoring function that measures the proportion of coverage is

$$\frac{d - a + 1}{d - c + 1}$$



6.4.3 Example of CeC Functionalities

A sample query received by CeC from AnC is given below. This query is the output from the AUI.

Listing 6.1: User Query

```
{RID: $Req_1$,
Domain: "Healthcare",
Subdomain: "Endocrinology",
attributes:["Patient's Age","Gender", "Diabetes Type", "Blood_Pressure", "Hba1c"],
quality_feature:{(Volume, 50), (Variety,"xml"), (Value, 0.8)}
weights:[0.1,0.1,1],
semantics:[1,0,1]}
}
```

MU of CeC starts the matching process by requesting the pair $(mdid, tagSet)$ of each metadata within the domain("Healthcare") and the subdomain("Endocrinology"). In response, PrC sends 4 pairs of $(mdid, tagSet)$ from the MTD to CeC.

Listing 6.2: Metadata Ids and Tagsets Received from PrC

```
# METADATA PROVIDER (1)
{id:$mdid_1$,
tagSet:{"Patient Age", "Gender", "Current Smoking Status", "Smoking Duration",
    "HbA1C"}}

# METADATA PROVIDER (2)
{id:$mdid_2$,
tagSet: {"Patient Age", "Gender", "Presence of Complication", "Presence of
    Diabetes", "Presence of Albuminuria"}}

# METADATA PROVIDER (3)
{id:$mdid_3$,
tagSet:{"Patient Number", "Diabetes Duration", "Total Metabolic Score"}}

# METADATA PROVIDER (4)
{id:$mdid_4$,
tagSet:{"Patient Number", "Patient Age", "Diabetes Type", "Blood Pressure",
    "Hba1c", "Hemoglobin"}}
```

Matching Process:

The set MTD_{FC} of FCA concepts created out of the four pairs in Listing 6.2 are shown below:

Listing 6.3: FCA Concepts

```
FCA-concepts =[($mdid_1$,"Patient Age"),($mdid_1$,"Gender"), ($mdid_1$,"Current Smoking
    Status"), ($mdid_1$,"Smoking Duration"), ($mdid_1$,"Hba1c"), ($mdid_2$,"Patient
    Age"),($mdid_2$,"Gender"), ($mdid_2$, "Presence of Complication"), ($mdid_2$,
    "Presence of Diabetes"), ($mdid_2$, "Presence of Albuminuria"), ($mdid_3$,"Patient
    ID"),($mdid_3$,"Diabetes Duration"), ($mdid_3$,"Diabetes Type"), ($mdid_3$,"Total
    Metabolic Score"), ($mdid_4$,"Patient ID"), ($mdid_4$,"Patient Age"),
    ($mdid_4$,"Diabetes Type"), ($mdid_4$,"Blood Pressure"), ($mdid_4$,"Hba1c"),
    ($mdid_4$,"Hemoglobin")]
```

The set of extents of the above FCA concepts are listed below:

$FCA\_Concept.extent([' PatientAge']) = objects[mdid_1, mdid_2, mdid_4]$

$FCA\_Concept.extent([' Gender']) = objects[mdid_1, mdid_2]$

$FCA\_Concept.extent([' DiabetesType') = objects[mdid_4]$

$FCA\_Concept.extent([' Blood_Pressure') = objects[mdid_4]$

$FCA\_Concept.extent([' Hba1c') = objects[mdid_1, mdid_4]$

The union of all the above extents is the set $result = \{mdid_1, mdid_2, mdid_4\}$.

For every $mdid \in result$ we calculate a "Value" that measures the semantic closeness between $attSet$ of the query and $tagSet$ (intent) of $mdid$ (extent) in MTD_{FC}.

$$mdid_1.value = 3/5 * 100 = 60\%$$

$$mdid_2.value = 2/5 * 100 = 40\%$$

$$mdid_1.value = 4/5 * 100 = 80\%$$

The set of pairs $\{(mdid_1, 60\%), (mdid_2, 40\%), (mdid_4, 80\%)\}$ is sent to RU, which after receiving this data requests PrC to fetch the full metadata descriptions corresponding to the metadata identifiers in the received set. This set of metadata is shown below.

Listing 6.4: Matching Metadata Fetched from PrC

```
# METADATA PROVIDER (1)
{mdid:mdid₁,
quality_feature:{(Volume, 80), (Variety,"cvs"), (Velocity,365), (Release
    Year,2021),(Reliability,high), (Safety,"low"), (Veracity,"not known"),
    (Value, ⋆)}}

# METADATA PROVIDER (2)
{
mdid:mdid₂,
quality_feature:{(Volume, 70), (Varietry,"xml"), (Velocity,120), (Release
    Year,2020), (Reliability,"medium"), (Safety,"medium"), (Veracity,"not
    known"), (Value,⋆)}}

# METADATA PROVIDER (4)
{
mdid:mdid₄,
quality_feature:{(Volume, 150), (Variety,"cvs"), (Velocity,7), (Release
    Year,2019), (Reliability,high), (Safety,"high"), (Veracity,"accepted"),
    (Value, ⋆)},
```

RU fills in the “Value” part in each metadata description. After this is done, we have the following result.

Table 6.1: Structuring Quality Features of Query in Listing 6.1

query_id	Volume	Variety	Velocity	Release Year	Availability	Reliability	Safety	Veracity	Value
q1	50	xml	0	0	0	0	0	0	0.8
Weights	0.1	0.1	0	0	0	0	0	0	1
Semantics	1	0	0	0	0	0	0	0	1

Table 6.2: Structuring Quality Features of Metadata in Listing 6.5

metadata	Volume	Variety	Velocity	Release Year	Availability	Reliability	Safety	Veracity	Value
mdid1	80	cvs	365	2021	0	5	1	0	0.6
mdid2	70	xml	120	2020	0	3	3	3	0.4
mdid4	150	cvs	7	2019	0	5	5	3	0.8

Listing 6.5: Matched Metadata with Values

```
{mdid1, quality_feature:{(Volume, 80), (Variety,"cvs"), (Velocity,365), (Release
    Year,2021),(Reliability,high), (Safety,"low"), (Veracity,"not known"),
    (Value, 60)}}\\
{mdid2,
quality_feature:{(Volume, 70), (Varietry,"xml"), (Velocity,120), (Release
    Year,2020), (Reliability,"medium"), (Safety,"medium"), (Veracity,"accepted"),
    (Value,40)}}\\
{mdid4,
quality_feature:{(Volume, 150), (Variety,"cvs"), (Velocity,7), (Release
    Year,2019), (Reliability,high), (Safety,"high"), (Veracity,"accepted"),
    (Value, 80)},
```

Ranking Process:

To start ranking, the quality features in user query and quality features in every metadats are structured to include all 9 quality features, as explained in Section6.4.1.

That is, all nine quality features are going to be included in the vector giving value 0 to the ones that are not included either in the metadata or in the query. Likewise, in the query, the values for weights and semantics of not included quality features is set to 0. This results in having query and metadata quality_feature vectors to have the same length, as shown in Table 6.1 and 6.2.

Scoring Process:

For “Variety” attribute which is of type “nominal”, the scoring function in Equation 2 is applied. For “Availability” attribute, one of the scoring functions given for “Interval” type is applied. In our example, “Availability” is not mentioned either in the query or in a metadata. For “Volume” and “Value” attributes, the semantics specified in the query is “MB”. So, the scoring function in Equation 4 is applied. The similarity measures calculated by using the function

$$\sum_{i=1}^{9} w_i \times score_i$$

is shown in Table 6.3.

Table 6.3: Generated Scores for Each Metadata

metadata	Volume	Variety	Velocity	Release Year	Availability	Reliability	Safety	Accuracy	Value	Score
mdid1	0.14	0	0	0	0.32	0	0	0	0.8	1.26
mdid2	0.13	0.1	0	0	0.48	0	0	0	0.5	1.21
mdid4	0.17	0	0	0	0.7	0	0	0	1	1.87

The list of metadata ranked according to decreasing similarity measure values is given in Listlisting 6.6. The $mdid_4$ is placed on top of the list as it has the highest rank value with respect to the query specifications.

Listing 6.6: Metadata After Ranking

```
# METADATA PROVIDER (4)
{user_id: Prov_3,
mdid:mdid_4,
Domain: "Healthcare",
Subdomain: "Endocrinology",
attributes:[("Patient ID", "PID"),("Patient Age". "age"),("Diabetes
    Type","DM.Type"),("Blood
    Pressure","BP"),("Hba1c","BG_avarage"),("Hemoglobin","HgB")],
quality_feature:{(size, 150), (format,"cvs,xml"), (update_frequency,"every 7
    days"), (release_year,2019), (reliability,high), (safety,"high"),
    (accuracy,"accepted"), (value, 60)},
contract:{
context_dsc:{origin:'Canada', ReleaseYear:2021, DS-URL:'www.mdid_4.com},
context_spc:{name:"Adil", contact:"adil@email.com", location:"xCountry"},
context_sdc:{Regions:['x','y'], Organizations:'', SD-Fee:''}
legal_rules:{}
}}

```

```
# METADATA PROVIDER (1)
{$user_{id}$: $Prov_1$,
mdid:$mdid_1$,
Domain: "Healthcare",
Subdomain: "Endocrinology",
attributes:[("Patient Age", "age"),("Gender","Sex"), ("Current Smoking
    Status","smk.status"), ("Smoking Duration","smk.duration"),
    ("Hba1c","BG.AVG")],
quality_feature:{(size, 80), (format,"cvs"), (update_frequency,"every 365 days"),
    (release_year,2021),(reliability,high), (safety,"low"), (accuracy,"not
    known"), (value, 40)},
contract:{
context_dsc:{origin:'Canada', ReleaseYear:2021, DS-URL:'www.$mdid_1$.com},
context_spc:{name:"Adil", contact:"adil@email.com", location:"xCountry"},
context_sdc:{Regions:['x','y'], Organizations:'', SD-Fee:''},
legal_rules:{}
}}

# METADATA PROVIDER (2)
$user_{id}$: $Prov_2$,
mdid:$mdid_2$,
Domain: "Healthcare",
Subdomain: "Endocrinology",
attributes:[("Patient Age", "p.age"),("Gender","p.gender"), ("Presence of
    Complication","complications"), ("Presence of Diabetes","diabetes_status"),
    ("Presence of Albuminuria","albumin")],
quality_feature:{(size, 70), (format,"xml"), (update_frequency,"every 120 days"),
    (release_year,2020), (reliability,"medium"), (safety,"medium"),
    (accuracy,"not known"), (value, 20)}
contract:{
context_dsc:{origin:'Canada', ReleaseYear:2021, DS-URL:'www.$mdid_2$.com},
context_spc:{name:"Adil", contact:"adil@email.com", location:"xCountry"},
context_sdc:{Regions:['x','y'], Organizations:'', SD-Fee:''}
legal_rules:{}
}}
```

Chapter 7

Complexity Analysis and Performance Analysis on a Case Study

In this chapter, the algorithmic complexity of the data search engine CeC is first discussed. Next, we give a case study to bring out the claims that we have made on the CeC performance. The case study uses an ontology of "Diabeties" sub-domain of "Healthcare" domain. The ontology includes 104 concept terms and phrases. These are used in the "Attribute" section of the metadata descriptions and in analyst's queries. The quality dimensions are the same as those described in earlier chapters.

7.1 Algorithmic Complexity of Dataset Search Engine

The algorithm that drives the Metadata directory construction in PrC and the algorithm that enables query construction in AnC are outside the scope of the dataset search engine. They may be viewed as "preprocessing" algorithms that prepare the input to the dataset search engine component CeC. Once these are readied, and the analyst query is input to CeC, the dataset search begins. So, we need to focus only on this "dynamic complexity" (cost), namely the cost of discovering and delivering the datasets for an input analyst query. This complexity is a measure of how long both the matching and the ranking algorithms

Table 7.1: Search Engine Input Sizes

Description	Size	Remarks
Query Tags	n_T	$\leq N_O$ (number of tags in the Ontology)
Query Quality	n_Q	Number of Quality Values (≤ 9)
User Semantics	n_Q	Number of Quality Values (≤ 9)
Weights	n_Q	Number of Quality Values (≤ 9)
Metadata Directory	$N_{d,sd}$	Number of Metadata Descriptions
Metadata Tags	n_{mdid}	Number of tags in the metadata with Id *mdid* ($\leq N_O$)
matchingSet	M_Q	Size of the set of matched metadata ($\leq N_{d,sd}$)
mdStore	M_Q	Same as size of *matchingSet* ($\leq N_{d,sd}$)
Metadata Model Size	MS_{mdid}	Size of metadata model whose Id is *mdid*

would take to complete and deliver the datasets. The two main factors that measure the complexity are *Time Factor* (TF) and *Space Factor* (SF). Both are measured as functions of input size to the algorithm.

AnC inputs to CeC the analyst query, and CeC communicates with PrC to fetch data as explained in Chapter 6. Because the query specifies one domain/sub-domain pair (d, sd), the metadata in the directory for this fixed (d, sd) pair will be accessed and processed. So, it is sufficient to consider the size of metadata directory for (d, sd) for complexity analysis. The analyst query has the semantic part that lists the query attributes, and the quality part (along with weights and semantics) that lists quality dimension values. So, for complexity analysis of one query we may formalize the inputs and their sizes as in the Table 7.1.

7.2 Complexity of Matching Phase

We give the steps of CeC actions during matching phase, and discuss the time and space complexity factors for each step.

- *Receiving and Saving Analyst Query:* Analyst query is received by CeC and saved in $queryVector$. The time, as well as space complexity of this step is $\mathcal{O}(|queryVector|)$. Because the vector of quality dimensions in the query is bounded by 9, the user semantics and weight vectors are also bounded by 9, and the tag vector size is bounded by N_O, the time and space complexity of this step is

$$\mathcal{O}(N_O) \tag{6}$$

- *Requesting and Receiving Metadata Tags:* CeC requests PrC for the set $tagsetPair = \{(mdid(x), tagSet(x))\}$, for $x \in MI[d, sd]$. In the metadata directory $MI[d, sd]$ there are $N_{d,sd}$ metadata descriptions, and the number of tags in the description of *mdid*-referring metadata is n_{mdid}. So, the total number of data elements copied by PrC and sent to CeC is

$$\sum_{mdid \in tagsetPair} (1 + n_{mdid})$$

  Because the number of tags in each metadata description is bounded by the number of leaf nodes in the ontology, we have $n_{mdid} \leq N_O$. Hence, the above expression is bounded by $N_{d,sd}(1 + N_O)$. In "big O" notation, we write this as $\mathcal{O}(N_{d,sd}) \times N_O)$. The received data is saved by RU of CeC in MTD storage. So, the time complexity (copying cost and file transmission cost) and the space complexity (local storage) are both equal to

$$\mathcal{O}(N_{d,sd}) \times N_O) \qquad (7)$$

- *Matching Using Formal Concept Analysis:* Assuming the pairs in MTD can be recast in FCA at unit cost (we need only consider the first element in each pair as a singleton set), we need to estimate the complexity of "set intersection operations" between each *tagSet* in MTD and the *attSet* part in user query. Using Python method (van Rossum, 2018), the runtime complexity of the *set.intersection() method* on sets with n and m elements is $\mathcal{O}(min(n, m))$, because we need to check whether each of the elements in the smaller set is a member of the larger set. The cost of checking membership for an element in a set is $\mathcal{O}(1)$, so the runtime complexity of intersection algorithm is $\mathcal{O}(min(n, m)) * \mathcal{O}(1) = \mathcal{O}(min(n, m)$. Assuming $|attSet| = x$, and $|tagSet(x)| = y_{mdid}$, the complexity of calculating one intersection is $\mathcal{O}(min(x, y_{mdid})$. The maximum value of $(min(x, y_{mdid}))$ is N_O, when both *attSet* and the *tagSet* of every *mdid* has all the leaf elements of the ontology. So, the worst case cost for one intersection calculation is N_O. To calculate the extent of all formal concepts during the matching phase, we need to repeat the above step for every $mdid \in MTD$. So, the total cost of matching phase is

$$\sum_{mdid \, \in \, MTD} (min(x, y_{mdid})) \, ,$$

which in "big O" notation is

$$\mathcal{O}(N_O^2 \times N_{d,sd}) \tag{8}$$

- *Complexity of Value Calculation:* For the sake of conceptual clarity, in Chapter 6 the value calculation was done after matching was completed. In the implementation, it is done during the "intersect calculation" stage of the matching phase. That is, if $tagSet \cap attSet \neq \emptyset$, then we calculate

  $$value(mdid) = \frac{|tagSet \cap attSet|}{|attSet|}$$

  Therefore, the additional costs (1) "one division" for every metadata selected by the matching algorithm, and (2) inserting $value_{mdid}$ in the $tagsetPair$. After inserting we replace the tuple $(mdid(x), tagSet(x)$ by the triple $(mdid, tagSet_{mdid}, value(mdid)$ in the data store *matchingId*. So, there is no additional space complexity. The additional time complexity is due to M_Q division operations. Because $M_Q \leq N_{d,sd}$, this additional complexity is

  $$\mathcal{O}(N_{d,sd}) \tag{9}$$

7.3 Complexity of Ranking Phase

For the sake of conceptual clarity we have shown *matchingId* in MU and *matchingSet* in RU as separate storage spaces. In the implementation, both MU and RU may share the same storage. So, there is no data transfer between MU and RU. There are three steps in the ranking phase. We explain them and give the complexity of each step.

- *RU requests PrC for the full metadata description:* For every *mdid* in a triple in *matchingSet*, RU requests PrC for the full metadata description. After receiving it, RU copies $value(mdid)$ from that triple to the "value field" in the quality dimensions list of the metadata received from PrC. Then, the metadata is saved locally in *mdStore*. The costs for these steps are itemized below. in the "cost" expressions, the k_is are constants:

- ◦ PrC copies one metadata model of size MS_{mdid} — cost $= k_1 \times MS_{mdid}$
- ◦ PrC transfers that file of size MS_{mdid} — cost $= k_2 \times MS_{mdid}$
- ◦ RU copies $value(mdid)$ in the metadata quality field and stores it in *mdStore* — cost $= k_3 \times MS_{mdid}$

So, the cost for one *mdid* is $K \times MS_{mdid}$, where $K = (k_1 + k_2 + k_3)$. The total cost of transferring and copying the metadata descriptions corresponding to every $mdid \in matchingSet$ is

$$\sum_{mdid \,\in\, matchingSet} (K \times MS_{mdid})) = \mathcal{O}(MM \times M_Q) = \mathcal{O}(MM \times N_{d,sd})) \tag{10}$$

where $MM = max_{mdid \in matchingSet}\{MS_{mdid}\}$, and the size M_Q of *matchingSet* is bounded by $N_{d,sd}$, the total number of metadata descriptions in the metadata directory.

- *Calculating Similarity Measures:* The quality vector in the given query will be compared with the quality vector in each metadata description stored in *mdStore* and their similarity measure will be calculated. We have standardized the quality vector size to be 9. To calculate the similarity for one pair of vectors, 9 scoring functions, one for each pair of attributes, will be evaluated. A scoring function evaluation requires a finite number (2 or 3) of simple comparisons and arithmetic operations. Therefore, the number of operations requited to calculate 9 scoring function is bounded by 27. The similarity calculation from these 9 scores requires the "weighted sum" of the scores. That requires 9 multiplications and 8 additions. To sum up, the total number of comparisons/arithmetic operations required to calculate the similarity measure of one pair of vectors is a constant, say α. So, for calculating the similarity measures between the query quality vector and every quality vector of the metadata model in *mdStore* the cost is $\alpha \times M_Q$ (size of *mdStore*), whose order of complexity is

$$\mathcal{O}(M_Q) = \mathcal{O}(N_{d,sd}) \tag{11}$$

- *Ranking Metadata Models:* The key for ranking the set of metadata in *mdStore* is the similarity measure calculated for each metadata model. Using Quicksort, the ranking time is

$$\mathcal{O}(N_{d,sd} \times \ln N_{d,sd}) \tag{12}$$

After ranking, CeC send the ranked metadata list to AnC, Because the size of the ranked metadata list is $\mathcal{O}(N_{d,sd})$, both the time and space complexity of this step is

$$\mathcal{O}(N_{d,sd}) \tag{13}$$

7.4 Total Complexity of Search Engine

Total time complexity is the sum of expressions in Equation 6, Equation 7, Equation 8, Equation 9, Equation 10, Equation 11, Equation 12, and Equation 13. It is given by the expression

$$\begin{aligned} TimeComp = \mathcal{O}(N_O) + \mathcal{O}(N_{d,sd} \times N_O) + \mathcal{O}(N_O^2 \times N_{d,sd}) + \mathcal{O}(N_{d,sd}) + \mathcal{O}(MM \times N_{d,sd}) + \\ \mathcal{O}(N_{d,sd}) + \mathcal{O}(N_{d,sd}) + \mathcal{O}(N_{d,sd} \times \ln N_{d,sd}). \end{aligned} \tag{14}$$

In general, the size N_O of ontology leaf nodes, and the number $N_{d,sd}$ of datasets published under (d, sd) are the only two “independent input parameters”. It is easy to infer that $M_Q \leq N_{d,sd}$. The maximum meta model size MM is $\mathcal{N_O}$, because the total size of “non-functional elements, and the number of contexts” in a meta model description is bounded by a constant. With these observations and using the algebra of Big O (Zeil, 2020), we simplify the expression in Equation 14 as follows. We observe that

$$\mathcal{O}(MM \times N_{d,sd}) = \mathcal{O}(N_O \times N_{d,sd})$$

because $MM \leq N_O$. The expressions $\mathcal{O}(N_O)$, $\mathcal{O}(N_{d,sd})$, and $\mathcal{O}(N_O \times N_{d,sd})$, are subsumed by $\mathcal{O}(N_O^2 \times N_{d,sd})$. With these two observations, the time complexity expression in

Equation 14 can be rewritten as

$$\mathcal{O}(N_O^2 \times N_{d,sd}) + \mathcal{O}(N_{d,sd} \times \ln N_{d,sd}) \tag{15}$$

Total space complexity is the sum of expressions in Equation 6 Equation 7 and Equation 10.

$$\mathcal{O}(N_O) + \mathcal{O}(N_{d,sd} \times N_O) + \mathcal{O}(MM \times N_{d,sd})$$

Because $MM \leq N_O$, and the expression $\mathcal{O}(N_O)$ is subsumed by other terms, we can rewrite the above cost as

$$\mathcal{O}(N_O \times N_{d,sd}) \tag{16}$$

We observe that for a fixed size ontology when the number of datasets increases, $N_{d,sd}$ is the dominating factor both for time and space complexity. With this assumption, space complexity is *linear* in the metadata directory size $N_{d,sd}$, and time complexity is dominated by *sorting* algorithm complexity $\mathcal{O}(N_{d,sd} \times \ln N_{d,sd})$. These measures are only the "worst case" complexities. In real-life datasets, we can expect a much better performance because the number of metadata descriptions that match a given query may be much less that $N_{d,sd}$. the prototype implementation reveals such scenario.

7.5 Case Study: Performance Evaluation

The evaluation focuses on two aspects. First, we focus on the results output from matching and ranking algorithms for a small metadata dataset $mdSet_1$ of size 10. The full descriptions of these 10 metadata are provided in Appendix C. We construct 12 queries to validate our claim of "completeness" of matching phase on $mdSet_1$. These 12 queries are constructed using the interface AUI of AnC. The first 10 that are constructed to have complete match are listed in listing 7.1. The next two queries that have only partial or no match are shown in listing 7.2. Terms in query q_{11} are not from the ontology and only one term of q_{12}, which is "*exercising hours/week*″, included in two metadata descriptions is from the ontology. Second, we run many experiments on "matching and ranking" by varying metadata set size, attribute size in metadata descriptions, and the query structure. We observe from the plotted graphs, the run time performance is as predicted by the theoretical measure.

7.5.1 Value Calculation and Validating Completeness of Matching

Listing 7.1: Query List

```
{request_id: $q_1$,
user_id:$analyst_1$
Domain: "Healthcare",
Subdomain: "Diabetes",
attributes:["patient age","patient gender", "Diabetes Type", "blood pressure",
    "Hba1c"],
quality_feature:{(volume, 150), (variety,"xml"), (velocity,365),
    (release_year,2014), (availability[2014,2023]), (reliability,3), (safety,3),
    (veracity,3), (value, 0.5)}
weights:[0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1],
semantics:[1,0,-1,0,0,1,1,1,1]
}
{request_id: $q_2$,
user_id:$analyst_1$
Domain: "Healthcare",
Subdomain: "Diabetes",
attributes:["Diabetes Type","blood pressure","Hba1c","metabolic score","diabetes
    duration","patient number"],
quality_feature:{(volume, 100), (variety,"xml"), (velocity,21),
    (availability,[2000,2022])}
weights:[1,0.5,0.3,0.9],
semantics:[1,0,-1,0]
}
{request_id: $q_3$,
user_id:$analyst_1$
Domain: "Healthcare",
Subdomain: "Diabetes",
attributes:["Hba1c","Metabolic Score","Diabetes Duration","patient number"],
quality_feature:{(volume, 50), (variety,"xml"), (veracity,3), (value, 0.8)}
weights:[1,0.5,0.3,0.9],
semantics:[1,0,1,1]
}
{request_id: q_4,
user_id:analyst_1
Domain: "Healthcare",
Subdomain: "Diabetes",
attributes:["patient number","Presence of metabolic disorder","Blood
    biochemistries-full blood count","Low-density lipoprotein (LDL)","Overall
    medication usage","Frequency of emergency department presentation"],
quality_feature:{(volume, 250), (velocity,180), (release_year,2020), (safety, 3)}
weights:[1,0.5,0.3,0.9],
semantics:[1,-1,1,1]
}
{request_id: q_5,
user_id:analyst_1
Domain: "Healthcare",
Subdomain: "Diabetes",
attributes:["patient number","Number of risk factors","Total metabolic
    score","Number of psychiatric conditions","patient age"],
quality_feature:{(volume, 150), (variety,"xml"), (velocity,60),
    (release_year,2021), (availability,[2010,2022]), (safety,1), (value, 0.5)}
weights:[1,0.3,0.5,0.3,0.1,0.1,0.9],
semantics:[1,0,-1,1,0,1,1]
}
{request_id: q_6,
user_id:analyst_1
Domain: "Healthcare",
Subdomain: "Diabetes",
attributes:["patient age","patient gender","Lifestyle patterns","sleep
    quality","Consumption of food","alcohol","Hba1c"]
quality_feature:{((velocity,60), (release_year,2021), (availability,[2010,2022]),
    (safety,1), (value, 0.5)}
weights:[0.5,0.3,0.1,0.1,0.9],
semantics:[-1,1,0,1,1]
}
```

```
{request_id: $q_7$,
user_id:$analyst_1$
Domain: "Healthcare",
Subdomain: "Diabetes",
attributes:["patient age","patient gender","Diabetes Type","exercising
    hours/week","blood sugar daily average","blood pressure","Hba1c"],
quality_feature:{(volume, 80), (availability,[2010,2022]), (safety,1), (value,
    0.6)}
weights:[0.1,0.1,0.1,0.9],
semantics:[1,0,1,1]
}
{request_id: $q_8$,
user_id:$analyst_1$
Domain: "Healthcare",
Subdomain: "Diabetes",
attributes:["patient age","patient gender","Diabetes Type","company of continuous
    glucose measurement (CGM)","cgm daily average","blood pressure","Hba1c","body
    mass index"],
quality_feature:{(volume, 100), (variety,"cvs"), (velocity,120),(value, 0.5)}
weights:[1,0.3,0.1,0.9],
semantics:[1,0,1,1]
}
{request_id: $q_9$,
user_id:$analyst_1$
Domain: "Healthcare",
Subdomain: "Diabetes",
attributes:["patient age","patient gender","body mass index","physical activity
    type","exercising hours/week","blood pressure","Hba1c"],
quality_feature:{(volume,120), (value, 0.75)}
weights:[0.3,0.9],
semantics:[1,1]
}
{request_id: $q_{10}$,
user_id:$analyst_1$
Domain: "Healthcare",
Subdomain: "Diabetes",
attributes:["patient age","physical activity type","exercising hours/week","blood
    pressure","Hba1c"],
quality_feature:{(volume, 50)}
weights:[1],
semantics:[1]
}
```

Listing 7.2: Extra Queries for Testing

```
    {request_id: q_11,
    user_id:analyst1,
    Domain: "Healthcare",
    Subdomain: "Diabetes",
    attributes:{age, blood sugar, life style},
    quality_feature:{},
    weights:[],
    semantics:[]
    }

    {request_id: q_12,
    user_id:analyst5,
    Domain: "Healthcare",
    Subdomain: "Diabetes",
    attributes:{age, exercising hours/week, life style},
    quality_feature:{}
    weights:[],
    semantics:[]
    }
```

Results of matching are shown in Table 7.2. The metadata that does not match the query is provided value 0. A non-zero value represents the relevance level to the query, which we use in ranking algorithm. We verified manually the correctness of the results. We also applied matching is on a larger metadata set $mdSet_3$ of size 50 and checked the results, provided in Appendix C. So, this experiment enables us to validate our claim that

Table 7.2: Value Calculated for Each Matching Metadata to Queries (q_1 *to* q_{12})

query_id	**mdid1**	**mdid2**	**mdid3**	**mdid4**	**mdid5**	**mdid6**	**mdid7**	**mdid8**	**mdid9**	**mdid10**
q1	0.6	0.4	0	0.6	0	0.2	0.6	0.8	0.8	0.6
q2	0.17	0	0.5	0.67	0.17	0.33	0.17	0.33	0.5	0.33
q3	0.25	0	0.75	0.5	0.25	0.5	0.25	0.25	0.25	0.25
q4	0	0	0.17	0.33	1	0.17	0	0	0	0
q5	0.2	0.2	0.4	0.2	0.2	1	0.2	0.2	0.2	0.2
q6	0.43	0.29	0	0.14	0	0.14	0.86	0.43	0.29	0.29
q7	0.43	0.29	0	0.43	0	0.14	0.43	0.86	0.57	0.57
q8	0.38	0.25	0	0.38	0	0.13	0.38	0.5	0.88	0.5
q9	0.43	0.29	0	0.29	0	0.14	0.43	0.57	0.57	0.86
q10	0.4	0.2	0	0.4	0	0.2	0.4	0.6	0.6	1
q11-(no correct term)	0	0	0	0	0	0	0	0	0	0
q12(one term correct)	0	0	0	0	0	0	0	0.33	0	0.33

Table 7.3: q_1 Quality Features Specifications

	Volume	Variety	Velocity	Release Year	Availability	Reliability	Safety	Veracity	Value
values	150	xml	365	2014	[2014,2023]	3	3	3	0.5
weights	0.5	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
Semantics	1	0	1	0	0	1	1	1	1

CeC matching method is sound (no relevant metadata is left out) and complete (all relevant metadata ate retrieved).

7.5.2 Validating Correctness of Ranking

Because queries $q_1, \cdots, q_{10}$ match every dataset in $mdSet_1$, with varying relevance value, all datasets in $mdSet_1$ are selected by matching phase. The next phase of CeC is to rank the datasets in $mdSet_1$. Because the quality dimensions in the queries will be different, we expect different ranked lists of $mdSet_1$. Below we show the result of ranking for two different queries and comment on their different ordering.

The quality features of q_1 and q_2 are shown in Table7.3 and 7.5. The main difference between them is that query q_1 gives the highest weight to "volume" and q_2 gives the highest weight to "velocity". Both quality dimensions have the semantics "MB". Ranking for q_1, provided in Table 7.4 shows that the algorithm ranked "higher" the metadata that provides the highest Volume value. Ranking for q_2 provided in Table 7.6 shows the metadata with the highest number of Velocity ranked higher. So, user semantics integrated in the scoring functions work as expected.

Table 7.4: Ranking Result for q_1

Scores	Metadata ID
1.91	mdid8
1.49	mdid2
1.47	mdid3
1.35	mdid7
1.20	mdid1
1.14	mdid6
1.14	mdid5
1.14	mdid10
1.13	mdid4
1.07	mdid9

Table 7.5: q_2 Quality Features Specifications

	Volume	Variety	Velocity	Release Year	Availability	Reliability	Safety	Veacity	Value
values	100	xml	21	2000	[2000,2022]	0	0	0	0.5
weights	0.1	0.1	0.8	0.1	0.1	0.1	0.1	0.1	0.1
Semantics	1	0	1	0	0	1	1	1	1

Table 7.6: Ranking Result for q_2

Scores	Metadata ID
2.15	mdid2
2.13	mdid1
2.00	mdid5
1.96	mdid9
1.67	mdid7
1.56	mdid3
1.47	mdid4
0.85	mdid8
0.72	mdid6
0.46	mdid10

Table 7.7: Matching Performance of Attribute-Based Experiment

	$mdSet_1(10)$	$mdSet_2(30)$	$mdSet_3(50)$	$mdSet_4(70)$	$mdSet_5(100)$
$q_1(5)$	0.0002	0.0002	0.0003	0.0005	0.0007
$q_2(10)$	0.0003	0.0005	0.0009	0.0011	0.0017
$q_3(20)$	0.0005	0.0008	0.0015	0.0019	0.0031
$q_4(30)$	0.0006	0.001	0.0016	0.0022	0.0035

7.5.3 Performance Evaluation of Matching Phase

In this section we focus on evaluating the performance of matching algorithm that includes the calculation of value. As there are different dimensions involved in this evaluation, different experiments are applied to understand the performance. These experiments are explained below.

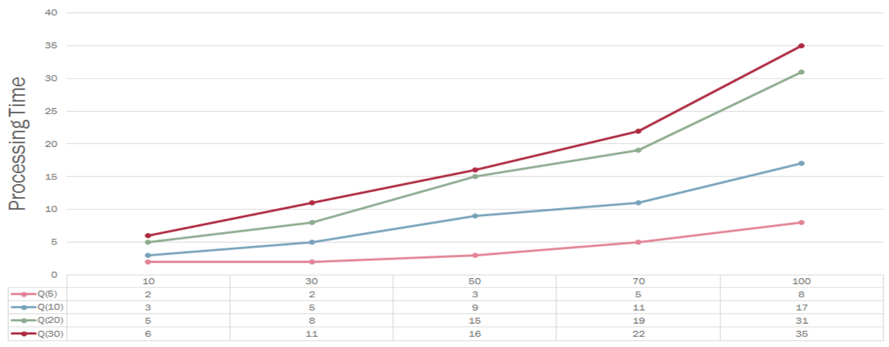
Attributes Based Experiment:

In this experiment we constructed four queries q_1, q_2, q_3, and q_4, with number of attributes 5 in q_1, 10 in q_2, 20 in q_3, and 30 in q_4. We applied the matching algorithm for each query on five metadata sets $mdSet_1$, $mdSet_2$, $mdSet_3$, $mdSet_4$, and $mdSet_5$ whose respective sizes are 10,30,50,70, and 100. For each metadat set $mdSet_i$ we let the attributes in the metadata descriptions to vary as follows:

- For the experiment of query q_1 the number of attributes in every $mdSet_i$ was allowed to vary in the ranged $[2, 8]$.
- For the experiment of query q_2 the number of attributes in every $mdSet_i$ was allowed to vary in the ranged $[5, 15]$.
- For the experiment of query q_3 the number of attributes in every $mdSet_i$ was allowed to vary in the ranged $[15, 25]$.
- For the experiment of query q_4 the number of attributes in every $mdSet_i$ was allowed to vary in the ranged $[25, 35]$.

We created a table and a graph to show the relative performance of matching phase for the above scenarios. These are shown in Table 7.7 and in Figure 7.2.

We also observed the extent of matching for every case. Our overall observation is for each query the runtime performance increases linearly in the size of metadata set, regardless



	10	30	50	70	100
Q(5)	2	2	3	5	8
Q(10)	3	5	9	11	17
Q(20)	5	8	15	19	31
Q(30)	6	11	16	22	35

Figure 7.1: Matching Performance of Attribute-based Experiment

Table 7.8: Matching Performance of a Random Query (2)

Number of Metadata	Matching Processing Time
10	0.0001
30	0.0004
50	0.0008
70	0.0012
100	0.0015

of the extent of matching and variation in attribute size in a metadata description.

Metadata Based Experiment:

This experiment does not consider the number of attributes in either metadata or query, and measures performance of matching a randomly generated query with 15 attributes on the five metadata sets $mdSet_i$, $i = 1, \cdots, 5$. Metadata sets where given a random number of attributes ranging in $[4, 70.$ to 70). The purpose is to observe whether our observation stated above, is valid. The results of this experiment are shown in Table 7.8 and in Figure 7.8. The performance results show “linear time increase” in the size of metadata.

General Matching Performance for all Queries Put Together:

In this experiment the purpose is to observe the “total run time behaviour” for all queries on each metadata set. So, we decided to match all queries in listing 7.1 on the five metadata sets $mdSet_1$, $mdSet_2$, $mdSet_3$, $mdSet_4$, and $mdSet_5$. We recall that their respective sizes are 10,30,50,70,100 metadata. The attributes of each metadata is randomly selected within the range $[4, 70]$. The results of this experiment are shown in Table 7.9, illustrated in Figure 7.3.

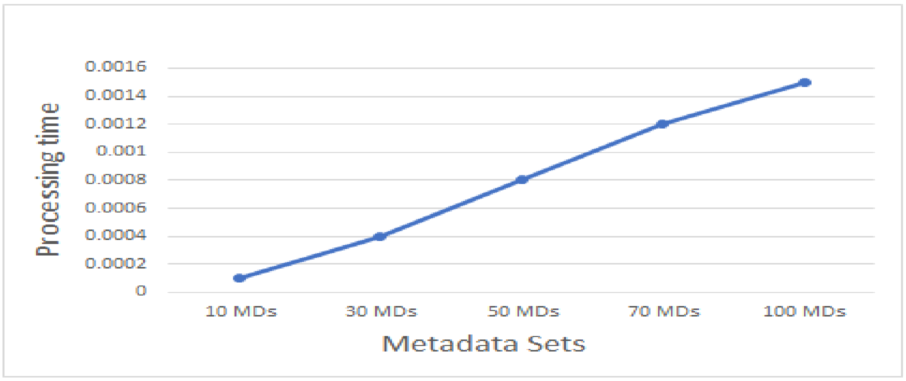


Figure 7.2: Matching Performance of a Random Query)

Table 7.9: Matching Performance - Average Case

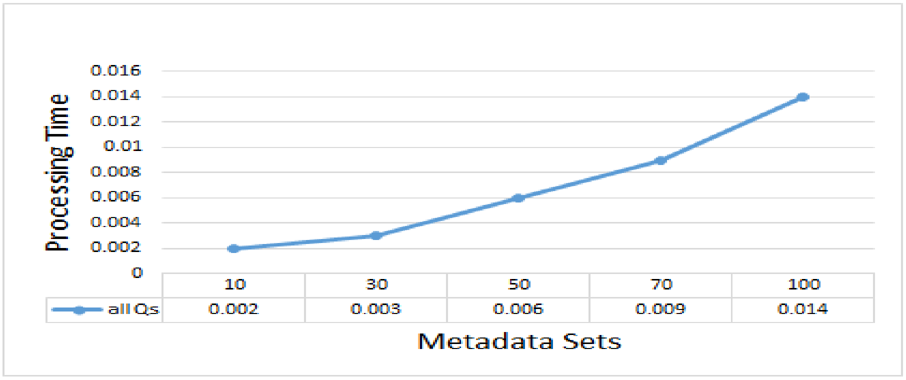
Metadata	Processing Time
10	0.002
30	0.003
50	0.006
70	0.009
100	0.014

7.5.4 Performance of Ranking Algorithm

Ranking is applied on fixed size "quality vectors" because we have restricted the number of quality dimensions to 9 and standardized the user semantics and weigh vectors to be of size 9. We calculated similarity measures using the scoring functions described in Chapter 6 on the sets $mdSet_1$, $mdSet_2$, $mdSet_3$, $mdSet_4$, and $mdSet_5$. Based on the similarity measures we ranked the datasets, in decreasing order of similarity measure. Results in Table 7.10 show the average performance of ranking the metadata sets.

Table 7.10: Ranking Performance

Queries	Number of Metadata	Average Processing time
Q1-Q10	10	0.003
Q1-Q10	30	0.008
Q1-Q10	50	0.014
Q1-Q10	70	0.020
Q1-Q10	100	0.026



	10	30	50	70	100
all Qs	0.002	0.003	0.006	0.009	0.014

Figure 7.3: Matching Performance - Average Case

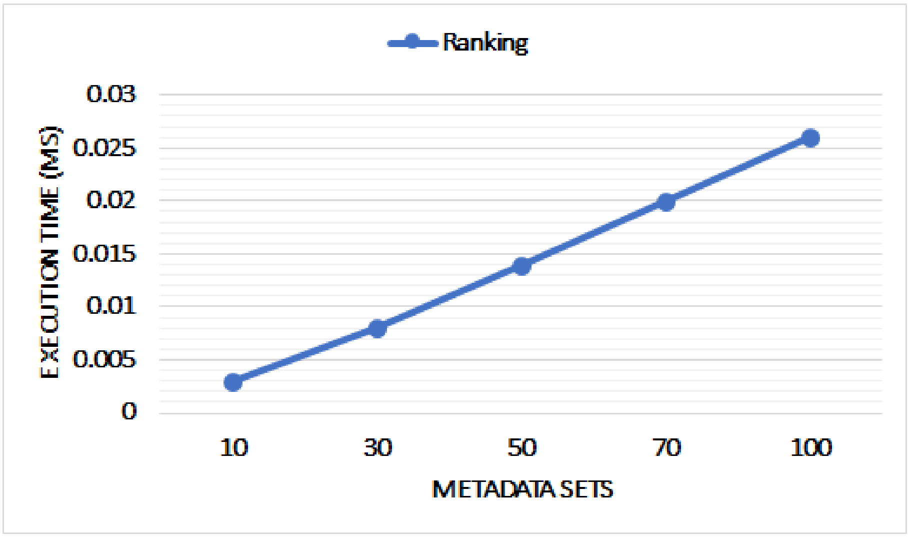


Figure 7.4: Performance of Ranking

7.6 Summary and Comparison

The prototype implementation now exists in the environment *Ubuntu*18.3 using Python 3.7. So, the absolute measurements might vary when the prototype is run under a different environment. However, the relative performance must be similar. In fact, this is conformed by our experiment under Ubuntu 18.3 and Windows 10 environments.

The prototype is only a "small example to show proof of concept". It illustrates that the TCF components work well as intended, and perform their tasks in a predictable and reasonably fast time frame. The data for experiments, although was varied, are of small size. In Chapter 8 we compare our work with previous works from three different levels that we used in TCF design, and mention future directions of research related to comparison of the performance of our dataset search engine CeC with other published methods.

Chapter 8

Conclusion and Future Work

This thesis has introduced a tight-coupling framework for the development of a dataset search engine that can be used by data practitioners when selecting a dataset that can be trusted for *semantic relevance*, *specified quality*, and *context-dependent usability.* The framework is implemented as a running prototype system, and its performance has been evaluated using datasets with varying size, and number of quality and semantic attributes.

The framework design is based on component-based software development method, and service-oriented paradigm. Metadata descriptions of datasets are modeled as services by dataset providers and are published. Data practitioners (analysts) are enabled to formulate their requests in a rich query structure that includes semantics, quality dimensions, and user-centric semantics. The dataset search engine discovers the datasets from metadata publications that "best match" the analyst request. Because, the dataset search engine component uses only the user-centric requests in the query for matching against published metadata, and for ranking the set of matched metadata, we call the framework "tightly-coupled". The above three tasks are encapsulated as three separate components whose design details can be updated independently. So, the component design is "loosely coupled". However, the combined behavior of the components will achieve *semantic relevance*, *specified quality*, and *context-dependent usability* in the discovered datasets. Below, we discuss how these claims and research goals stated in Chapter 1 are met by the results of this thesis. During this discourse we give an assessment of our solutions, and bring out the merits of our approach as compared to the related work discussed in Chapter 2. Finally, we identify the directions of future work to extend and generalize the work completed in this thesis.

8.1 Meeting The Goals

In achieving the goals stated in Chapter 1 we positioned ourselves with respect to (1) current concerns on lack of right methodologies for metadata descriptions, (2) the need for tight coupling between dataset providers and data seekers, (3) discovery based on user-centric semantic and quality concerns, and (4) contextual compatibility. We identified "dataset provider", "dataset requester", and "the engine that tightly couples them" as three entities for dataset discovery process, and encapsulated them in three components. This design methodology has enabled us to use appropriate models to describe each component, and develop suitable methods to fulfill their functionalities. We shall explain in Section 8.2 how this design can be extended and generalized to make the dataset search more robust. None of the published work on dataset search engine (Bogatu et al., 2020; Castelo et al., 2021; Koutras et al., 2021; Noy et al., 2019) has explained the design methodology on which their search engines were developed. Below we explain how well the other goals are met using this design.

8.1.1 DSC Formation - Metadata Modeling and Metadata Directory

The term "DSC" introduced in Chapter 1 comprehensively refers to the "functional information and context" pertaining to datasets at a source. In our design, PrC functionality has fulfilled this goal. The service-oriented modeling of metadata in PrC is a new contribution to Big Data area. As reviewed earlier, to the best of our knowledge there exists no previous work on metadata modeling. The service model that we have introduced has the following significance:

- It is a "formal" model, although we have not used any formal specification notation to describe its parts. The structuring and precise descriptions of metadata are kept simple in order for it to be "understandable" for data practitioners. We claim "formality" only in the way the syntax of its parts are defined, and the way we have explained the semantics of its parts. All the essential elements suggested in the experience reports (Koesten et al., 2020) are included in describing the metadata. With the syntactic separation and semantics that we use there is no ambiguity in metadata description.

- The context information of DSC is made part of “contract part” in the metadata model, separated from the functional elements of the model. The significance of this design is that the service function can be varied without varying the contract and vice versa. Metadata descriptions retrieved by matching a query semantics can be *shared* by many whose local contexts vary while all such contexts remain *compatible* to the contractual context specifications. That is, contextual compatibility and data shareability become easier to fulfill.

- By including the quality features under service part and separating it from non-functional properties of metadata, our design has enabled a selected number of quality features as “mandatory” to be specified in every metadata description while allowing the dataset provider specify any number of non-functional attributes to enhance understandability and usability of metadata. In addition, the set of quality features and the set of non-functional features have nothing in common which allows updating one set without affecting the other.

- In our design, the information for a metadata model is collected from the dataset provider. As such, the “completeness” of information in it is fully dependent on the information input by the dataset provider. In order to achieve a level of completeness desired by the service provider, we designed the user interface PUI through which a service provider, supported by OnC, can interactively and iteratively input metadata information. The system creates separate files for the different parts metadata service model, saves them in a storage pool, and uploads it in the metadata directory only when the service provider requests the “submission of metadata to the metadata directory”. In PUI design we have the provision to allow service providers edit separate parts of metadata information and save. So, the design provides the service providers achieve their desired level of completeness in metadata descriptions. The information in the metadata service model is also “complete” in another sense, namely the different components in TCF use every piece of information in the metadata model to fulfill their goals. That is, “no information” in the metadata model is left unused. This claim is illustrated in Table 8.1.

- In the current design, the OnC support is assumed to exist. Moreover, in it there exists one ontology to support the semantics of concept terms for all datasets under one $(domain, sub - domain)$ pair. As part of future work in Section 8.2 we explain how to generalize ontology support to improve the robustness of queries and matching process.

- The metadata model is both "scalable" and "reusable". It is scalable because many other parts describing the metadata can be added as well as information within each part can be expanded. The reusability aspect applies to individual parts of the model. For example, it may be possible to *compose* metadata descriptions in order to describe "composition of datasets". The semantic support for compositions will come from the OnC generalization. In Section 8.2 we discuss reusability aspect in some detail.

Table 8.1: Metadata Information Utilization in TC Framework

Service	Functional	PreCondition	PrC
		PostCondition	PrC
		Attributes	CeC
	Non Functional	Title	AnC
		Description	AnC
		Data Source	AnC
		Others	AnC
	Non Functional	Quality Features	CeC
Contract	Context	SPC, DSC, SDC	AnC
	Legal Rules		AnC

8.1.2 DRC Formation - Query Modeling and Dataset Selection

The term "DRC" introduced in Chapter 1 comprehensively refers to the role of an analyst in query construction and final dataset selection. This goal is fulfilled in Chapter 5. As brought out in Chapter 2, Auctus search engine (Castelo et al., 2021) is more complex and powerful than Google dataset search engine (Noy et al., 2019). The functionality of the user interface https://auctus.vida-nyu.org/ in Auctus is rich enough to allow users upload their own datasets, use keywords and different constraints such as date range and geographical area to compose queries to search for new datasets. The interface https://datasetsearch.research.google for Google dataset search engine has very limited options to choose for requesting datasets. Both Google and Auctus do not provide a query construction model for analysts in which quality features, concept terms, and user semantics and preferences can be stated. We provide a rich query structure for analysts which includes all the above options. We have created ART (Analyst Request Template), a knowledge model of the GQM model (Basili, 1992; R. V. Solingen & Berghout, 1999), and a user interface supported by it in order to assists analyst compose rich queries. Analysts need not know the GQM model in order to construct their queries, because the user interface AUI assists them with ART. The current ART has limitations, in terms of "suggesting goal-oriented" examples to identify a wide set of attributes. The design at present gives a "proof of concept" on which the examples and case study have been given in the thesis. In Section 8.2 we explain how to build a more robust "expert system" that can assist analysts to construct rich queries when faced with discovering real-world datasets.

In the current design, the dataset selection in AnC is based on the analyst's local context and is left as a "manual process". No other published work considered dataset selection based on contextual constraints. We let the analyst team follow Delphi method to get the right to review, filter, and re-request better matched metadata in order to best suit their contextual constraints. This process may be iteratively continued until the analyst team gets their most relevant datasets. So, we need to automate this part, with a formal approach to represent contextual constraints and define contextual compatibility. Formalizing the notion of "contextual compatibility", and automating the process of selecting metadata that fit the local context of analyst are left to future work explained Section 8.2.

8.1.3 DSC - DRC Tight Coupling: Dataset Search Engine

The dataset engine in CeC matches DRC with DSC and ranks the matched metadata sets. Because of the "multi-dimensional" nature and "heterogeneous types" involved in describing the quality items we have, we decided to extract datasets that semantically match a given query and then rank the set of extracted metadata set. That is, we use *semantics-based matching* and *similarity-based* ranking. The rationale is the following:

- To increase "relevance" of datasets, they must match the semantics in the query, otherwise they do not match the analysis goal of the analyst. In the metadata model every metadata description has the "Attributes" part in which concept terms from the OnC ontology are included as "tags" to provide semantics for attributes. Every query structure has a set of attributes which are “tags” extracted from the Local Knowledge Unit in PrC. So, we can perform semantic matching between a query and metadata. We have used the basic principles of FCA for semantic matching. The primary reason is that FCA is formal, and because of the “maximal” property of the result it produces we are assured of the “completeness” of the matching result. That is, all those that semantically match are extracted by FCA method. The other reason for using FCA in the current design is that it has the expressive power and tools when we generalize the ontology support in OnC, and either allow semantic terms at any level of a Ontology or allow semantic terms from more than one ontology to be chosen for metadata description. In Section 8.2 we discuss these aspects.

- The values of quality features specified in a query may not “exactly” match the value of quality features specified in a metadata description. Also, the set of quality features specified in a query may not exactly match the set of quality features used in metadata description. So, we decided to go for “best matching” between them, and let the user specify modes, semantics, and preferences for their notion of “best matching”. Based on the quality feature specifications, we calculate a measure of similarity between a user query and a metadata description. We use the similarity measures as keys to rank the set of metadata that have been matched semantically.

8.1.4 Merits of TCF - A Summary of Comparison

Based upon the above summary we compare TCF features with other dataset search engine methods on three levels, as shown in Table 8.2. It shows “how well” we have met our goals in developing a dataset search engine whose features are new and novel, as compared with existing dataset search engines.

The levels of comparison are data requester level (analyst), the dataset search engine level, and the dataset provider level. In published works, as we showed in the review in Chapter 2, the role and responsible activities of agents at these levels have not been discussed in full. So, we show in Table 8.2 a full comparison on the level-based criteria dataset discovery engine to justify the overall merits of TCF dataset search engine.

Table 8.2: Merits of TCF: Level Based Criteria of Dataset Discovery

	Publication Year	2022	2021	2021	2020	2019	2018
	Method Name	TCF	Valentine	Auctus	Dataset in Data Lake	Google Dataset Search	Maritime
Provider Level	Context Based	Y					
	Attribute description of Dataset (Part of Metadata (MD))	Y			Y-possible		
	Contextual Contract (Part of Metadata)	Y					
	Wizard Based Metadata Publication	Y					
	Metadata provided by DSP owners	Y					
	MD generated by programs	NA		Y		Schema.org	
	Quality Features of DS	Y					Y
	Flexibility (in choice of MD elements)	Y					
Analyst Level	Primary Data Collection	Y			Y		
	Delphi (Trusted Authority)	Possible					Y
	User Interface	Y-Rich		Y-Limited		Y-Limited	
	Context Based	Y					
	Attribute Based Querying	Y			Y-possible		
	Keywords Based Search	Possible	Y	Y		Y	
	Quality Features	Y					Y
	Analyst Requirement Template (ART)	Y-GQM					Delphi
	Filtering	Y-Manual		Y		Y	
	ART Querying Structure	Rich		Limited			
Matching and Ranking Level	Metadata Based	Y		Y	Y	Y	
	Model Based	Y					
	Ontology Based	Y				Not Known	
	Matching	Semantics Similarity (FCA)			N- Numerical Similarity		
	User Centric Semantics	Y					
	Multi Featured Quality Ranking	Y					
	Quality-Based Similarity Assessment	Y					

8.2 Future Work

Research on designing dataset search engines that are robust and efficient is still in its infancy. We have compared the dataset search engine proposed in this thesis with others to bring out its merits and advantages. Still, TCF is only a “prototype proof of concept”. It can and must be improved in several directions.

- *Reuse of Components:* Component CeC in our design is reusable “as is” in many application domains, including “social networks” (for forming trusted networks), and “web services” (ranking services and products). It is only essential to have consistency of syntax/semantics for query-specified quality values and quality values in the system records. The ontology support of OnC can be used in other domain-related applications. It is possible to reuse components with modifications. For example, in order to process large dynamic datasets we need to reconstruct and reorganize concept terms from the ontology. The generic approach for efficient redesign of Galois (concept) lattices proposed in (Valtchev et al., 2002) has the potential for efficiently updating the existing metadata models for efficient reuse. This research will be challenging, interesting, and fruitful for reuse of dynamically changing datasets.

- *Generalize Tags:* The current design uses only the leaf nodes of one ontology in OnC. We can relax this limitation in two stages.
  Stage 1: We can let concept terms in any level of ontology to be used as tags. That is, dataset providers can use concept terms higher up in the ontology in specifying the metadata attributes in the metadata model. This when recorded in Local Knowledge Component of PrC will allow analysts to choose any concept term from PrC. The current set structure for recording must be modified to reflect the *partial order relation*, as in the ontology, of concept terms. Moreover, the matching unit in CeC must now have this relation in order to integrate it with FCA formalism algorithms (Poelmans et al., 2013; Valtchev et al., 2002) to define all formal concepts and continue with the matching and ranking algorithms.
  Stage 2: We can let more than one ontology for each $(domain, subdomain)$ pair. In order to allow providers and analysts use concept terms from any level of any ontology, it is necessary to merge the ontologies. A future study here is to make use of the

research done in "combining ontology" (Porello & Endriss, 2011; Stumme & Maedche, 2001) to enrich OnC support to service providers through the use of multiple ontologies in creating metadata tags. In particular, the FCA-based approaches (Stumme & Maedche, 2001; Valtchev et al., 2002) look interesting and have the potential to fit in with the FCA analysis that we have done for matching. another promising approach. If "compound concepts", composed from concept terms are to be used, then the most promising approach seems to follow the generalized pattern extraction method for concept lattices (Kwuida et al., 2014).

- *Expert Support for ART:* In the current design, the ART in AnC is the "expert knowledge of GQM" that assists analysts to compose rich queries. ART support for "attribute" selection part is somewhat limited and can be enriched by providing a large number of questions and suggested answers to them in the "attribute segment" section of ART. Building an expert system for this purpose is a time consuming task. The work (Chen, Homayoun, & Wang, 2003) seems a viable first step to investigate how we can design an intelligent question-answering system for attribute selection in a given ($domain - subdomain$). An intelligent processing will require (1) representing knowledge about attributes, their types, and concepts associated with them, and (2) a collection of questions/answers in the forms of rules for selection. A formal representation of knowledge and reasoning based on it will give ART a robust support.

- *Automate Contextual Compatibility and Final Selection of Datasets:* Every analyst has a local context in which the dataset delivery can be requested. If the analyst is a member of an organization which has many research labs in different locations, then more than one delivery context might be compatible. Thus, local analyst contexts are essential to get datasets that can be used in local contexts. We foresee the automation of this step will require a formal definition of "compatibility between contexts", and a separate "context toolkit". After some preliminary investigation we found that context compatibility can be defined below:

  Context Compatibility: It is a *partial order* relation, denoted $\sqsubseteq$ on a finite collection of contexts. For two contexts C_1 and C_2 in a given collection, define $C_1 \sqsubseteq C_2$ if "C_2 contains all the information of C_1". That is, C_1 is compatible with C_2 if it is a "sub-context" of context C_2. In practice, we must assume that the given collection of contexts are based on "the same set of dimensions and the domains of values associated with the dimensions" (Alsaig et al., 2019), and contexts C_1 and C_2 are elements of such a collection. A result in context calculus (Alsaig, 2022; Alsaig et al., 2019) states that "if a property (predicate) p is true in context C_1, and $C_1 \sqsubseteq C_2$, then p is true in C_2.

In (Alsaig, 2017) a user interface is given to define dimensions, types, and domain of values for dimensions. This tool also provides an environment in which contexts defined in the environment can be manipulated using a fully defined context algebra. So, a challenging future work is to export the above tool as a component, and create interfaces of it with AnC and PrC in order to automate contextual compatibility validation. This extension will make our current design more robust.

- *Enriching OnC Structure:* In real world situations, all dataset providers may not be using the same ontology. As examples, more than one ontology has been used in medical science (Hastings, Ceusters, Jensen, Mulligan, & Smith, 2012; Larsen & Hastings, 2018) to represent mental functioning and psychiatric disorders. They have explained with simple examples how to relate terms across the ontologies. So, it is demonstrated that ontologies for a specific $(domain, subdomain)$ pair will have some commonalities. In such situations, combining two ontologies becomes a challenging issue. We already discussed this issue above in Stage 2 of "Generalize Tags".

  Another direction of enriching OnC support is to investigate the inclusion of *Trusted Authority* (TA), realized as as an "intelligent agent" who can validate and certify the quality measures that service providers will include in metadata descriptions. Delphi method was used in (Strozyna et al., 2018) for assessment of identified datasets according to a set of defined quality criteria. They have used a questionnaire with a list of sources and the experts were asked to assign a mark to each quality criterion using a four level rating scale "High, Medium, Low, Not Known". The paper (Strozyna et al., 2018) says that "experts were given some information and statistics on the datasets", but does not specify on the type of information given. When we are dealing with a large number of datasets, the manual approach of Delphi can be replaced by the "intelligent agent" system. Following their basic Delphi idea, the intended functionality of the "intelligent agent" is to accept the metadata description and statistical information on the dataset as input and certify a "degree of acceptance" of the quality dimension values specified in metadata descriptions. The "intelligent system" may issue a certificate showing the "degree of acceptance" and the service provider might include it in metadata description. This certificate may be used as an "additional criterion" for dataset selection by an analyst.

- *Query-Similarity Based Knowledge Discovery:* We propose a direction of research on *Contextual Reasoning on Metadata Databases* which can be viewed as a *Knowledge-based System* on which "learning" and "reasoning" are possible. The basic idea is to create a "Warehouse" component WhC with interface to AnC, and store in it metadata sets discovered for different user queries, and for several contexts for a query Ganter and Kuznetsov (2001). That is, for each query Q_i, $i = 1, \cdots, N$, we construct the pairs $(C_{ij}, MetList_{ij})$, for $j = 1, \cdots N_i$, where $MetList_{ij}$ is the ranked list of metadata filtered for context C_{ij} from the list of metadata received from CeC for query Q_i. The idea is to regard this collection as a contextualized knowledge-based rule-based system and use Contelog (Alsaig, 2022) for retrieving metadata that satisfy contexts that are all mutually compatible or/and those metadata that match quality attributes (not their actual values) of several queries. This research will provide ML (Machine Learning) and Deep Learning communities the ability to reason on the warehouse of data with their metadata, contexts, and queries corresponding to different datasets and select metadata that best fit their data analysis goals. Currently, AI community needs such targeted data. With their powerful tools, they can discover meaningful information and insights from already used queries and contexts that might be useful in other "compatible contexts". Information such as common interests (for analysts), universal concerns (over many contexts), and interesting research topics (different domain/subdomains) might give the AI community the kind of data they need.

Appendix A

Glossary

- **ACI:** AnC to CeC Interface
- **AI:** Artificial Intelligence
- **AnC:** Analyst Component
- **ANT:** Analysis Team
- **API:** AnC to PrC Interface
- **ART:** Analysis Request Template
- **AUC:** Authentication Component
- **AUI:** Analyst User Interface
- **BD:** Big Data
- **CAI:** CeC to AnC Interface
- **CeC:** Coupling Engine Component
- **CPI:** CeC to PrC Interface
- **domSet:** Domain Set
- **DQ:** Data Quality
- **DRC:** Data Requirement Context

- **DSC:** Dataset Context
- **DSE:** Data Search Engine
- **DSP:** Dataset Provider
- **EHR:** Electronic Health Record
- **EM:** Exact Match
- **EUI:** Expert User Interface
- **FCA:** Formal Concept Analysis
- **GQM:** Goal Question Metrics
- **LB:** Less is Better
- **LKC:** Local Knowledge Component
- **MB:** More is Better
- **mdid:** Metadata ID
- **MI:** Metadata Index
- **ML:** Machine Learning
- **MTD:** Metadata Directory
- **MTDC:** Metadata Directory Component
- **MU:** Matching Unit
- **NA:** Not Applicable
- **O:** Open Data Source
- **OEI:** Ontology to Expert Interface
- **OnC:** Ontology Component
- **OPI:** OnC to PrC Interface

- **OR:** Open Data Source with Registration
- **PA:** Paid Data Source
- **PAI:** PrC to AnC Interface
- **POI:** PrC to OnC Interface
- **PPA:** Data Source with Partial Paid Access
- **PR:** Private Data Source
- **PrC:** Provider Component
- **PUI:** Provider User Interface
- **QoD:** Quality of Data
- **RU:** Ranking Unit
- **SDC:** Service Delivery Context
- **SF:** Space Factor
- **SPC:** Service Provider Context
- **subdomSet:** Subdomain Set
- **TCF:** Tight Coupling Framework
- **TF:** Time Factor

Appendix B

Diabetes Ontology

In this appendix, we provide the the list of concept terms and phrases which are the leaf nodes of diabetes ontology Seng et al. (2021). These terms are used in the construction of the 10 metadata sets used Case Study of Chapter 7.

(1) adherence to anti-diabetic agents

(2) age at t2dm diagnosis

(3) alcohol

(4) alcohol consumption

(5) blood biochemistries/full blood count

(6) blood pressure control

(7) body mass index

(8) cardio metabolic

(9) characteristics of t2dm

(10) cigarettes

(11) citizenship

(12) cognitive status

(13) comorbidities of patients

(14) consumption of food

(15) continent of study

(16) current smoking status

(17) diagnosed at age

(18) dietary factors

(19) dose of anti-diabetic agents

(20) duration of t2dm

(21) echocardiographic variables

(22) employment status

(23) ethnicity

(24) exercise

(25) exercise length

(26) fatigue

(27) frailty

(28) frequency of emergency department presentation

(29) frequency/week

(30) glucose-dependent insulinotropic polypeptide (gip) and glucagon like peptide-1(glp-1)

(31) groningen intelligence test

(32) hba1c level

(33) hba1c variability

(34) health related quality of life

(35) hospital admission

(36) household income

(37) illness perception

(38) immigrant

(39) level of literacy

(40) low-density lipoprotein (ldl)

(41) marital status

(42) mean fasting plasma glucose levels

(43) medically attended hypoglycemia

(44) microaneurysm turnover and for the centralretinal thickness

(45) microvascular diseases

(46) morningness-eveningness

(47) non-immigrant

(48) number of anti-hypertensive agents

(49) number of cardiovascular risk factors

(50) number of psychiatric conditions

(51) number of risk factors

(52) overall medication usage

(53) patient address

(54) patient age

(55) patient gender

(56) patient number

(57) patient population

(58) patterns

(59) patterns of depression

(60) perceived self-efficacy

(61) perceived social support

(62) personality traits differences

(63) physical activity type

(64) physical functional status

(65) presence / levels of islet cells auto-immuneantibodies levels

(66) presence and severity of depression and anxiety symptoms

(67) presence of albuminuria

(68) presence of atherosclerosis

(69) presence of chronic kidney disease

(70) presence of diabetic nephropathy

(71) presence of diabetic retinopathy

(72) presence of hypertension

(73) presence of metabolic disorder

(74) presence of resistance hypertension

(75) presence of sleep disturbance

(76) race

(77) reasons for not participating in diabetes related program

(78) severity of albuminuria

(79) severity of anxiety symptoms

(80) severity of chronic kidney disease

(81) severity of diabetes distress

(82) severity of diabetes related complications

(83) severity of diabetic kidney disease

(84) severity of diabetic neuropathy

(85) severity of pain

(86) severity of psychiatric symptoms

(87) sleep quality

(88) smoking duration

(89) socio-economic status of patients

(90) status of t2dm at cancer diagnosis

(91) study design

(92) subtypes of t2dm

(93) total metabolic score

(94) trends and trajectories

(95) type of healthcare utilization

(96) type or specialty of care provider

(97) types of anti-diabetic agents used

(98) variability

(99) variability of fasting plasma glucose levels

(100) verbal intelligence

(101) waist circumference

(102) weight

(103) weight change over time

(104) year of study

Appendix C

Metadata and Calculated Value

C.1 Full 10 Metadata Description

In this appendix, we provide the description of full 10 metadata used for part of Case Study of Chapter 8.

Listing C.1: Metadata Description List

```
# METADATA PROVIDER (1)
     {user_id: $Prov_1$,
     mdid:$mdid_1$,
     domain: "Healthcare",
     subdomain: "Endocrinology",
     md_functional:{
     precondition:[$cond_1$:regsiteredRequester=true],
     postcondition:[$cond_1$:requesterFeedback()=true],
attributes:{
   names:['p.age', 'p.gender', 'p.smk.status', 'smk.duration', 'DM.AVG' ],
   tags:['patient age', 'patient gender', 'current smoking status', 'smoking
       duration', 'hba1c'],
   types:['numerical', 'string', 'boolean', 'numerical', 'numerical'],
   isKey:[0,0,0,0,0],
   description:["","others indicates not specified by the patient","smoking or
       not","in years","3 months blood sugar average"],
   completness:[100,90,100,100,100]}}
```

```
      nonfunctionalPart:{
      [
    title:"smoking influence on diabetic patients",
    description:"ooooo",
    data_source:"companyX",
    quality_features:{Volume:60, Variety:'xml', Velocity:365, Release Year:1990,
        Availability: [1990,2010], Reliability:3, Safety:3, Veacity:3, Value:0.6}
          contextual_contract:{
              spc:[name:xxx, ali, SP.Location:"XXX_MTL",
                  sp.contact:"xxx@email.com"],
              dsc:[origin:"montreal", releaseYear:2020, URL:"www.companyX.org"],
              sdc:[region:"c1,c2", sd-organization:"GEOxx,MED12", SD-Fee:"100
                  CAD/2GB"
       legalRules: ["Data can be used for educational and research purposes",
           "Dataset provider retains the right to amend data and users will be
           alerted when amendments are made"]
          }}
  # METADATA PROVIDER (2)
      {user_id: $Prov_2$,
       mdid:$mdid_2$,
       domain: "Healthcare",
       subdomain: "Endocrinology",
       md_functional:{
           precondition:[$cond_1$:regsiteredRequester=true],
           postcondition:[$cond_1$:requesterFeedback()=true],
           attributes:{
       nmaes:['age', 'sex', 'complications', 'albumenuria', 'diabetes'],
       tags:['patient age', 'patient gender', 'presence of complication',
           'presence of albumenuria', 'presence of diabetes'],
       types:['numerica', 'string', 'boolean', 'boolean', 'boolean']
       isKey:[0,0,0,0,0],
       description:["","","","",""],
       completeness:[90,90,90,90,90]}}
      nonfunctionalPart:{
          title:"Diabeties-Related complications",
    description:"xxx_yyy",
     datas_ource:"clinic_dep_x",
     quality_features:{Volume:120, Variety:'xml', Velocity:180, Release
         Year:2014, Availability: [2014,2023], Reliability:3, Safety:5,
         Veacity:3, Value:0.4}
      contextual_contract:{
          spc:[name:mmm, ayden, SP.Location:"YY_MTL",
              sp.contact:"ayden@email.com"],
          dsc:[origin:"canada", releaseYear:2020, URL:"www.clinicx.ca"],
          sdc:[region:"c1,c2", sd-organization:"GEOxx,MED12", SD-Fee:"100
              CAD/2GB"
    legalRules: ["Dataset provider retains the right to amend data and users
        will be alerted when amendments are made"]
          }}
  # METADATA (3)
      {user_id: $Prov_2$,
       mdid:$mdid_3$,
       domain: "Healthcare",
       subdomain: "Endocrinology",
       md_functional:{
           precondition:[$cond_1$:regsiteredRequester=true],
           postcondition:[$cond_1$:requesterFeedback()=true],
           attributes:{
       names:['metab.scr', 'diab.length', 'p.no'],
       tags:['metabolic score', 'dabetes duration', 'patient number'],
       types:['numerical', 'numerical', 'numerical'],
       isKey:[0,0,1],
       description:["metabolism score","number of years patient is diabetic",""],
       completeness:[50,90,100]}}
       nonfunctionalPart:{
             title:"changes of metabolica score of diabetec patients",
        description:"oooooeeee",
        datas_ource:"data_x",
        quality_features:{Volume:150, Variety:'cvs', Velocity:30, Release
            Year:2000, Availability: [2000,2029], Reliability:5, Safety:5,
            Veacity:3, Value:0}
      contextual_contract:{
          spc:[name:frank, SP.Location:"YY_MTL", sp.contact:"frank@email.com"],
          dsc:[origin:"canada", releaseYear:2020, URL:"www.dataX.org"],
          sdc:[region:"", sd-organization:"", SD-Fee:""]
    legalRules: ["Data can be used for educational and research purposes"]
      }}

  # METADATA (4)
      {user_id: $Prov_3$,
       mdid:$mdid_4$,
       domain: "Healthcare",
       subdomain: "Endocrinology",
       md_functional:{
           precondition:[$cond_1$:regsiteredRequester=true],
           postcondition:[$cond_1$:requesterFeedback()=true],
           attributes:{
        names:['no.', 'metabolic.disord', 'BP', 'HB', 'DM.type', 'a1c'],
        tags:['patient number', 'presence of metabolic disorder', 'blood
            pressure', 'hemoglobin', 'diabetes type', 'hba1c'],
        types:['string', 'boolean', 'numerical', 'numerical', 'string',
            'numerical'],
        isKey:[1,0,0,0,0,0],
        description:["","Y=1,No=0","","blood red cells percentage","t1,t2","blood
            sugar avg"],
        completeness:[100,80,95,90,90,90]}}
       nonfunctionalPart:{
           title:"diabetes influence on metabolica score",
        description:"oooooeeee",
        datas_ource:"clinic_fake",
        quality_features:{Volume:120, Variety:'xls', Velocity:30, Release
            Year:2016, Availability: [2016,2022], Reliability:1, Safety:5,
            Veacity:3, Value:0.6 }
       contextual_contract:{
           spc:[name:rayan, SP.Location:"YY_MTL", sp.contact:"ray@email.com"],
           dsc:[origin:"canada", releaseYear:2020, URL:"www.clifake.org"],
           sdc:[region:"c1,c2", sd-organization:"GEOxx,MED12", SD-Fee:"100
               CAD/2GB"
     legalRules: ["Data can be used for educational and research purposes"]
       }}


   # METADATA (5)
      {user_id: $Prov_4$,
       mdid:$mdid_5$,
       domain: "Healthcare",
       subdomain: "Endocrinology",
       md_functional:{
           precondition:[$cond_1$:regsiteredRequester=true],
           postcondition:[$cond_1$:requesterFeedback()=true],
     attributes:{
        names:['P_No', 'P_metabolic_disorder', 'B_bio_count', 'LDL', 'med_usage',
            'emergency_visit_count'],
        tags:['patient number', 'presence of metabolic disorder', 'blood
            biochemistries-full blood count', 'low-density lipoprotein (LDL)',
            'Overall medication usage', 'frequency of emergency department
            presentation'],
        types:['string', 'boolean', 'numerical', 'numerical', 'string',
            'numerical'],
        isKey:[1,0,0,0,0,0],
        description:["","P=1,N=0","","","",""],
        completeness:[100,100,100,100,100,100]}}
       nonfunctionalPart:{
           title:"survey on diabetic patients",
     description:"oooooeeee",
     data_source:"companyUU",
     quality_features:{Volume:115, Variety:'xls', Velocity:180, Release
         Year:2000, Availability: [2000,2020], Reliability:1, Safety:5,
         Veacity:6, Value:0}
      contextual_contract:{
           spc:[name:rose, SP.Location:"YY_MTL", sp.contact:"rose@email.com"],
           dsc:[origin:"canada", releaseYear:2020, URL:"www.uuc.org"],
           sdc:[region:"r10,r22", sd-organization:"MED12", SD-Fee:"100 CAD/2GB"
     legalRules: ["Dataset provider retains the right to amend data and users
         will be alerted when amendments are made"]
       }}
   # METADATA (6)
      {user_id: $Prov_4$,
       mdid:$mdid_6$,
       domain: "Healthcare",
       subdomain: "Endocrinology",
       md_functional:{
           precondition:[$cond_1$:regsiteredRequester=true],
           postcondition:[$cond_1$:requesterFeedback()=true],
           attributes:{
        names:['patient_no', 'no_risk_fact', 'score_meta', 'no_pyscha_cond',
            'patient_age'],
        tags:['patient number', 'number of risk factors', 'total metabolic
            score', 'number of psychiatric conditions', 'patient age'],
        types:['string', 'numerical', 'numerical', 'numerical', 'numerical',
            'numerical'],
        isKey:[1,0,0,0,0],
        description:["","","","",""],
        completeness:[100,100,100,100,100]}},
       nonfunctionalPart:{
           title:"Rask factors of diabetic patients",
     description:"ooooooeeee",
     data_source:"Hos_YX",
     quality_features:{Volume:80, Variety:'xml', Velocity:7, Release Year:2016,
         Availability: [2016,2023], Reliability:3, Safety:1, Veacity:3, Value:0.2}
       contextual_contract:{
         spc:[name:lina, SP.Location:"YY_MTL", sp.contact:"lina@email.com"],
         dsc:[origin:"canada", releaseYear:2020, URL:"www.hos_YX.org"],
         sdc:[region:"c1,c2", sd-organization:"GEOxx,MED12", SD-Fee:"100
             CAD/2GB"
     legalRules: ["Data can be used for educational and research purposes"]
       }}

   # METADATA (7)
      {user_id: $Prov_5$,
       mdid:$mdid_7$,
       domain: "Healthcare",
       subdomain: "Endocrinology",
       md_functional:{
           precondition:[$cond_1$:regsiteredRequester=true],
           postcondition:[$cond_1$:requesterFeedback()=true],
           attributes:{
        names:['p_age', 'p_sex', 'lifestyle', 'food_consump', 'alcohol', 'a1c'],
        tags:['patient age', 'patient gender', 'lifestyle patterns', 'consumption
            of food', 'alcohol', 'hba1c'],
        types:['string', 'string', 'string', 'numerical', 'boolean', 'numerical'],
        isKey:[0,0,0,0,0,0],
        description:["","","","",""],
        completeness:[100,100,60,80,80,100]}},
       nonfunctionalPart:{
           title:"diabetic patients life style",
     description:"projectxX in ddd",
     data_source:"collceted",
     quality_features:{Volume:300, Variety:'cvs', Velocity:365, Release
         Year:2005, Availability: [2005,2015], Reliability:5, Safety:5,
         Veacity:0, Value:0.6}
       contextual_contract:{
           spc:[name:yang, SP.Location:"YY_MTL", sp.contact:"yang@email.com"],
           dsc:[origin:"canada", releaseYear:2010, URL:"www.proXX.com"],
           sdc:[region:"", sd-organization:"", SD-Fee:""
     legalRules: ["Data can be used for educational and research purposes."]
```

```
      }}


# METADATA (8)
     {user_id: $Prov_6$,
      mdid:$mdid_8$,
      domain: "Healthcare",
      subdomain: "Endocrinology",
      md_functional:{
         precondition:[$cond_1$:regsiteredRequester=true],
         postcondition:[$cond_1$:requesterFeedback()=true],
         attributes:{
       names:['age', 'gender', 'dmType', 'exercise_h/w', 'dailyAvg', 'a1c'],
       tags:['patient age', 'patient gender', 'diabetes type', 'exercising
           hours/week', 'blood sugar daily average', 'hba1c'],
       types:['numerical', 'string', 'string', 'numerical', 'numerical',
           'numerical'],
       isKey:[0,0,0,0,0,0],
       description:["","","diabetes type","number of exercising hours per
           week","daily blood sugar average","3 months blood sugar average"],
       completeness:[100,100,100,90,60,100]}},
      nonfunctionalPart:{
         title:"diabetic patients life style",
    description:"projectxX in ddd",
    data_source:"collceted",
    quality_features:{Volume:400, Variety:'xml', Velocity:7, Release Year:2012,
        Availability: [2012,2020], Reliability:3, Safety:5, Veacity:6,
        Value:0.8},
      contextual_contract:{
         spc:[name:nora, SP.Location:"YY_MTL", sp.contact:"nora@email.com"],
         dsc:[origin:"canada", releaseYear:2010, URL:"www.proXX.com/nora"],
         sdc:[region:"", sd-organization:"", SD-Fee:""
    legalRules: ["Data can be used for educational and research purposes."]
      }}

```

```
  # METADATA (9)
    {user_id: $Prov_7$,
     mdid:$mdid_9$,
     domain: "Healthcare",
     subdomain: "Endocrinology",
     md_functional:{
         precondition:[$cond_1$:regsiteredRequester=true],
         postcondition:[$cond_1$:requesterFeedback()=true],
         attributes:{
      names:['age', 'dmType', 'CGM company', 'CGM_Avg/Day', 'BP', 'a1c', 'BMI'],
      tags:['patient age', 'diabetes type', 'company of continuous glucose
          measurement (CGM)', 'cgm daily average', 'blood pressure', 'hba1c',
          'body mass index'],
      types:['numerical', 'string', 'string', 'numerical', 'numerical',
          'numerical', 'numerical'],
      isKey:[0,0,0,0,0,0,0],
      description:["","","diabetes type","name of sensor compsny","daily blood
          sugar average reported by sensors","patient bp","3 months blood sugar
          average","patient bmi"],
      completeness:[100,100,70,90,60,100]}},
     nonfunctionalPart:{
         title:"accuracy of cgm products used for diabetic patients",
   description:"projectpp in stand",
   data_source:"collceted",
   quality_features:{Volume:100, Variety:'xls', Velocity:180, Release
       Year:2006, Availability: [2006,2019], Reliability:5, Safety:1,
       Veacity:6, Value:0.8},
     contextual_contract:{
         spc:[name:Jacob, SP.Location:"YY_MTL", sp.contact:"jacob@email.com"],
         dsc:[origin:"canada", releaseYear:2010, URL:"www.proPP.com"],
         sdc:[region:"", sd-organization:"", SD-Fee:""
   legalRules: ["Data can be used for educational and research purposes",
       "Dataset provider retains the right to amend data and users will be
       alerted when amendments are made"]
     }}
```

```
# METADATA (10)
     {user_id: $Prov_8$,
      mdid:$mdid_{10}$,
      domain: "Healthcare",
      subdomain: "Endocrinology",
      md_functional:{
          precondition:[$cond_1$:regsiteredRequester=true],
          postcondition:[$cond_1$:requesterFeedback()=true],
          attributes:{
       names:['age', 'BMI', 'activities_type', 'exercise/week', 'BP', 'a1c'],
       tags:['patient age', 'body mass index', 'physical activity type',
           'exercising hours/week', 'blood pressure', 'hba1c'],
       types:['numerical', 'string', 'string', 'numerical', 'numerical',
           'numerical'],
       isKey:[0,0,0,0,0,0],
       description:["","","diabetes type(t1,t2)","number of exercising hours per
           week","daily blood sugar average","3 months blood sugar average"],
       completeness:[100,100,100,90,60,100]}},
      nonfunctionalPart:{
          title:"diabetic patients life style",
    description:"projectxX in ddd",
    data_source:"collceted",
    quality_features:{Volume:130, Variety:'xml', Velocity:7, Release Year:2001,
        Availability: [2001,2021], Reliability:5, Safety:5, Veacity:0,
        Value:0.6},
      contextual_contract:{
          spc:[name:Jack, SP.Location:"YY_MTL", sp.contact:"jack@email.com"],
          dsc:[origin:"canada", releaseYear:2010, URL:"www.proXX.com/jack"],
          sdc:[region:"", sd-organization:"", SD-Fee:""
     legalRules: ["Data can be used for educational and research purposes",
         "Dataset provider retains the right to amend data and users will be
         alerted when amendments are made"]
      }
      }
```

C.2 Calculated Values for 50 Metadata

Table C.1: Calculated Value of Matching Metadata to Queries($q_1 to q10$)$mdSet_5 = 50$

Query#	metadataID	value	Query#	metadataID	value	Query#	metadataID	value	Query#	metadataID	value
q1	mdid1	0.60	q4	mdid6	0.17	q6	mdid10	0.29	q8	mdid4	0.38
q1	mdid10	0.60	q4	mdid34	0.17	q6	mdid14	0.29	q8	mdid44	0.25
q1	mdid14	0.20	q4	mdid48	0.33	q6	mdid16	0.14	q8	mdid49	0.25
q1	mdid2	0.40	q4	mdid25	0.17	q6	mdid17	0.14	q8	mdid6	0.13
q1	mdid33	0.20	q4	mdid28	0.17	q6	mdid18	0.14	q8	mdid7	0.38
q1	mdid4	0.60	q4	mdid4	0.33	q6	mdid2	0.29	q8	mdid8	0.50
q1	mdid44	0.40	q4	mdid18	0.17	q6	mdid24	0.14	q8	mdid9	0.88
q1	mdid49	0.40	q4	mdid12	0.17	q6	mdid27	0.14	q9	mdid33	0.14
q1	mdid6	0.20	q4	mdid5	1	q6	mdid29	0.14	q9	mdid19	0.14
q1	mdid7	0.60	q4	mdid50	0.17	q6	mdid31	0.14	q9	mdid21	0.14
q1	mdid8	0.80	q4	mdid3	0.17	q6	mdid33	0.14	q9	mdid1	0.43
q1	mdid9	0.80	q4	mdid49	0.17	q6	mdid37	0.14	q9	mdid34	0.14
q2	mdid33	0.17	q4	mdid17	0.17	q6	mdid4	0.14	q9	mdid14	0.14
q2	mdid3	0.50	q4	mdid40	0.17	q6	mdid44	0.29	q9	mdid4	0.29
q2	mdid1	0.17	q5	mdid1	0.2	q6	mdid45	0.14	q9	mdid8	0.57
q2	mdid6	0.33	q5	mdid10	0.2	q6	mdid48	0.14	q9	mdid15	0.14
q2	mdid7	0.17	q5	mdid11	0.2	q6	mdid6	0.14	q9	mdid44	0.14
q2	mdid49	0.50	q5	mdid12	0.2	q6	mdid7	0.86	q9	mdid49	0.14
q2	mdid5	0.17	q5	mdid15	0.6	q6	mdid8	0.43	q9	mdid6	0.14
q2	mdid16	0.17	q5	mdid16	0.4	q6	mdid9	0.29	q9	mdid10	0.86
q2	mdid10	0.33	q5	mdid18	0.4	q7	mdid10	0.57	q9	mdid7	0.43
q2	mdid9	0.50	q5	mdid2	0.2	q7	mdid14	0.14	q9	mdid43	0.14
q2	mdid50	0.17	q5	mdid23	0.2	q7	mdid2	0.29	q9	mdid9	0.57
q2	mdid8	0.33	q5	mdid3	0.4	q7	mdid33	0.14	q9	mdid18	0.14
q2	mdid11	0.17	q5	mdid32	0.2	q7	mdid4	0.43	q9	mdid37	0.14
q2	mdid4	0.67	q5	mdid33	0.4	q7	mdid44	0.29	q9	mdid2	0.29
q2	mdid38	0.17	q5	mdid37	0.2	q7	mdid49	0.29	q10	mdid1	0.40
q2	mdid15	0.17	q5	mdid38	0.2	q7	mdid6	0.14	q10	mdid10	1.00
q2	mdid44	0.50	q5	mdid4	0.2	q7	mdid7	0.43	q10	mdid18	0.20
q2	mdid32	0.17	q5	mdid44	0.4	q7	mdid8	0.86	q10	mdid2	0.20
q3	mdid3	0.75	q5	mdid45	0.2	q7	mdid9	0.57	q10	mdid34	0.20
q3	mdid10	0.25	q5	mdid47	0.2	q8	mdid1	0.38	q10	mdid4	0.40
q3	mdid50	0.25	q5	mdid49	0.2	q8	mdid10	0.50	q10	mdid43	0.20
q3	mdid4	0.50	q5	mdid5	0.2	q8	mdid14	0.13	q10	mdid44	0.20
q3	mdid1	0.25	q5	mdid50	0.2	q8	mdid15	0.13	q10	mdid49	0.20
q3	mdid9	0.25	q5	mdid6	1	q8	mdid19	0.13	q10	mdid6	0.20
q3	mdid5	0.25	q5	mdid7	0.2	q8	mdid2	0.25	q10	mdid7	0.40
q3	mdid6	0.5	q5	mdid8	0.2	q8	mdid21	0.13	q10	mdid8	0.60
q3	mdid8	0.25	q5	mdid9	0.2	q8	mdid33	0.13	q10	mdid9	0.60
q3	mdid7	0.25	q6	mdid1	0.43	q8	mdid37	0.13			

References

Akman, V., & Surav, M. (1996). Steps toward formalizing context. *AI magazine*, *17*(3), 55.

ALEGION. (2019, September). *Artificial intelligence and machine learning projects are obstructed by data issues* (Tech. Rep.). https://cdn2.hubspot.net/hubfs/3971219/Survey: Dimensional Research.

Alsaig, A. (2013). *Semantic-based, multi-featured ranking algorithm for services in service-oriented computing, master of computer science thesis* (Unpublished master's thesis). Concordia University, Montreal, Canada, http://spectrum.library.concordia.ca/978006/.

Alsaig, A. (2017). *Contelog engine and documentation: A prototype environment for reasoning with contexts.* http://www.contelog.com. Retrieved from `http://www.contelog.com`

Alsaig, A. (2022). *Contelog: A formal declarative framework for contextual knowledge representation and reasoning* (Unpublished doctoral dissertation). Concordia University.

Alsaig, A., Alagar, V., Mohammad, M., & Alhalabi, W. (2017). A user-centric semantic-based algorithm for ranking services: design and analysis. *Service Oriented Computing and Applications*, *11*(1), 101-120.

Alsaig, A., Alagar, V., & Ormandjieva, O. (2018a). A critical analysis of the v-model of big data. In *2018 17th ieee international conference on trust, security and privacy in computing and communications/ 12th ieee international conference on big data science and engineering (trustcom/bigdatase)* (p. 1809-1813). doi: 10.1109/TrustCom/BigDataSE.2018.00273

Alsaig, A., Alagar, V., & Ormandjieva, O. (2018b). Foundational issues on big data science and engineering. In *2018 ieee 16th intl conf on dependable, autonomic and secure computing, 16th intl conf on pervasive intelligence and computing, 4th intl conf on big data intelligence and computing and cyber science and technology congress(dasc/picom/datacom/cyberscitech)* (p. 995-1000). doi: 10.1109/DASC/PiCom/DataCom/CyberSciTec.2018.00141

Alsaig, A., Alagar, V., & Shiri, N. (2019). Formal context representation and calculus for context-aware computing: 7th eai international conference, iccasa 2018, and 4th eai international conference, ictcc 2018, viet tri city, vietnam, november 22–23, 2018, proceedings. In (p. 3-13). Informatik Rechnernetze: Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering. doi: 10.1007/978-3-030-06152-4_1

Alsaig, A., Mohammad, M., & Alsaig, A. (2015). Enhancing wearable systems by introducing context-awareness and fca. In (p. 264-271). doi: 10.1007/978-3-319-29236-6_26

Alsaig, A., Mohammad, M., & Alsaig, A. (2016). Enhancing wearable systems by introducing context-awareness and fca. In P. C. Vinh & V. Alagar (Eds.), *Context-aware systems and applications* (pp. 264–271). Cham: Springer International Publishing.

Ashmore, B., Calinescu, R., & Paterson, C. (2015). Assuring the machine language lifecycle: Desiderata, methods, and challenges. *ACM Comouting Surveys*, *54*(5).

Basili, V. (1992). *Software modeling and measurement, computer science technical report seriwes, cs-tr-2956 (umiacs -tr-92-96)* (Tech. Rep.). College Park, MD: University of Maryland.

Basili, V., & Weiss, D. M. (1984). A methodology for collecting valid software engineering data. *IEEE Transactions on Software Engineering*, 728–738.

Batini, C., Cappiello, C., Francalanci, C., & Maurino, A. (2009). Methodologies for data quality assessment and improvement. *ACM Computing Surveys*, *4*(3), 16:1–16:52.

Baum, D. (2020). *Cloud data lakes for dummies, second edition.* John Wiley & Sons, New Jersey, U.S.A.

Bawazir, A., Alhalabi, W., Mohamed, M., Sarirete, A., & Alsaig, A. (2018). A formal approach for matching and ranking trustworthy context-dependent services. *Applied Soft Computing*, *73*, 306–315.

Bian, J., Lyu, T., Loiacono, A., Viramontes, T. M., Lipori, G., Guo, Y., ... others (2020).

Assessing the practice of data quality evaluation in a national clinical data research network through a systematic scoping review in the era of real-world data. *Journal of the American Medical Informatics Association*, *27*(12), 1999–2010.

Blum, A. L., & Langley, P. (2018). Selection of relevant features and examples in machine learning. *Artificial Intelligence*, *97*, 245–271.

Bogatu, A., Fernandes, A. A., Paton, N. W., & Konstantinou, N. (2020). Dataset discovery in data lakes. In *2020 ieee 36th international conference on data engineering (icde)* (pp. 709–720).

Bourne, P. E. (2015). The nih big data to knowledge (bd2k) initiative. *J. Am. Med. Inform. Assoc.*, *22*, 1114.

Caldiera, V. R. B. G., & Rombach, H. D. (1994). The goal question metric approach. *Encyclopedia of software engineering*, 528–532.

Cappuzzo, R., Papotti, P., & Thirumuruganathan, S. (2020). Creating embeddings of heterogeneous relational datasets for data integration tasks. In *Proceedings of sigmod'20* (pp. 1–15).

Castelo, S., Rampin, R., Santos, A., Bessa, A., Chirigati, F., & Freire, J. (2021). Auctus: a dataset search engine for data discovery and augmentation. *Proceedings of the VLDB Endowment*, *14*(12), 2791–2794.

Chan, K. S., Fowls, J. B., & Weiner, J. P. (2010). Electronic health records and the reliability and validity of qualitative measures: a review of the literature. *Medcare Res Rev*, *67*(5), 503–527.

Chandrasekaran, D., & Mago, V. (2021, feb). Evolution of semantic similarity—a survey. *ACM Comput. Surv.*, *54*(2). Retrieved from `https://doi.org/10.1145/3440755` doi: 10.1145/3440755

Chapman, A., Simperl, E., Koesten, L., Konstantinidis, G., Ibáñez, L.-D., Kacprzak, E., & Groth, P. (2020). Dataset search: a survey. *The VLDB Journal*, *29*(1), 251–272.

Chen, T., Homayoun, B., & Wang, Y. (2003). Development of an intelligent agent-based gqm software measurement system. In *Proceedings of ats03* (pp. 188–197).

Cimiano, P., Hotho, A., & Staab, S. (2005). Learning concept hierarchies from text corpora using formal concept analysis. *Journal of artificial intelligence research*, *24*, 305–339.

Do, H. H., & Rahm, E. (2002). Coma: A system for flexible combination of schema matching approaches. In *Proceedings of the 28th vldb conference* (pp. 1–12).

Dong, X., Saha, B., & Srivastava, D. (2012). Less is more: Selecting sources wisely for integration. *Proceedings of the VLDB Endowment*, *6*(2), 37–48.

Eppler, M. J. (2006). *Managing information quality: Increasing the value of information in knowledge-intensive products and processes.* Springer-Verlag, Berlin Heidelberg.

ESS. (2014). *Ess handbook for quality reports.* ec.europa.eu/eurostat/web/european-statistical-system. (Accessed on 2022-06-15)

Fernandez, R. C., & et al, E. M. (2018). Seeping semantics: Linking datasets using word embeddings for data discovery. In *Proceedings of the 34th international conference on data engineering* (pp. 1–12).

Ganter, B., & Kuznetsov, S. O. (2001). Pattern structures and their projections. In *Conceptual structures: Broadening the base: 9th international conference on conceptual structures, iccs 2001 stanford, ca, usa, july 30–august 3, 2001 proceedings 9* (pp. 129–142).

Ganter, B., & Wille, R. (2012). *Formal concept analysis: mathematical foundations.* Springer Science & Business Media.

Ghaemi, A. (2011). *From domain models to components-a formal transformation approach towards dependable software development* (Unpublished doctoral dissertation). Concordia University.

Grime, M. M., & Wright, G. (2016). Delphi method. *Wiley StasRef: Statistics Reference Online*, 1–6.

Gumbrecht, J., & Fox, M. (2020). *Two coronavirus studies retracted after questions emerge about data.* Cable News Network. Retrieved from `https://www.cnn.com/2020/06/04/health/retraction-coronavirus-studies-lancet-nejm/index.html` (Accessed on 2022-05-24)

Harispe, S., Sánchez, D., Ranwez, S., Janaqi, S., & Montmain, J. (2014). A framework for unifying ontology-based semantic similarity measures: A study in the biomedical domain. *Journal of Biomedical Informatics*, 38–53.

Hastings, J., Ceusters, W., Jensen, M., Mulligan, K., & Smith, B. (2012). Representing mental functioning: Ontologies for mental health and disease. In *Proceedings of third international conference on biomedical engineering* (pp. 1–5).

Heineman, C. T., & Councill, W. T. (2001). *Component-based software engineering: Putting the pieces together.* Addison-Wesley Publishing Company.

Hsu, C. C., & Sandford, B. A. (2007). The delphi technique: making sense of consensus. *Practical assessment, research, and evaluation*, *12*(1), 1–10.

Ibrahim, N. I. (2012). *Specification, composition and provision of trustworthy context-dependent services* (Unpublished doctoral dissertation). Concordia University.

J, M., Bernstein, P. A., & Rahm, E. (2001). Generic schema matching with cupid. In *Proceedings of the 27th vldb conference* (pp. 1—10).

Jain, R. (2015, November). *Ai = contextual reasoning + learning* (Tech. Rep.). https://ngs.ics.uci.edu/ai-contextual-reasoning-learning: University of California, Irvine.

Joshi, N. (2017). *To 5 sources of big data.* https://www.allerin.com/blog/top-5-sources-of-big-data. (Accessed on 2018-10-15)

Khan, M. G., Callahan, T. J., Barnard, J., & et al. (2016). A harmonized data quality assessment terminology and framework for the secondary use of electronic health record data. *eGEMs-Generating Evidence & Methods to improve patient outcomes*, *4*(1), 1–21.

Knight, L. (2021). *Why you should be using gqm to measure software quality and value.* Retrieved from `https://www.connectall.com/why-you-should-be-using-gqm-to-measure-software-quality-and-value/` (Accessed on 2022-05-23)

Koesten, L., Simperi, E., Blount, T., Kacprzak, E., & Tennison, J. (2020). Everything you always wanted to know about a dataset: Studies in data summarisation. *International Journal on Human-Computer Studies*, *135*, 1–21.

Kornegay, C., & Segal, J. B. (2013). Selection of data sources. In *Developing a protocol for observational comparative effectiveness research: A user's guide* (p. 121-135). Agency for Healthcare Research and Quality (US), Rockville (MD).

Koutras, C., Siachamis, G., Ionescu, A., Psarakis, K., Brons, J., Fragkoulis, M., … Katsifodimos, A. (2021). Valentine: Evaluating matching techniques for dataset discovery. In *2021 ieee 37th international conference on data engineering (icde)* (pp. 468–479).

Kwuida, L., Missaoui, R., Balamane, A., & Vaillancourt, J. (2014). Generalized pattern extraction from concept lattices. *Annals of Mathematics and Artificial Intelligence*, *72*(1), 151–168.

Larsen, R. R., & Hastings, J. (2018). From affective science to psychiatric disorder: Ontology as a semantic bridge. *Frontiers in Psychiatry*, *9*(487), 1-13.

Li, Y., Gao, J., Meng, C., Li, Q., Su, L., Zhao, B., . . . Han, J. (2016). A survey on truth discovery. *ACM Sigkdd Explorations Newsletter*, *17*(2), 1–16.

Melnyk, S., Molina, H. C., & Rahm, E. (2002). Similarity flooding: A versatile graph matching algorithm and its application to schema matching. In *Proceedings of the 18th international conference on data engineering* (pp. 1—12).

Mihaila, G., Raschid, L., & Vidal, M. E. (2000). Using quality of data metadata for source selection and ranking. In *Proc. of webdb 2000* (pp. 1–6).

Mohammad, M., & Alagar, V. (2012). A component-based development process for trustworthy systems. *Journal of Software Evolution and Process (Special Issue: Papers on Processes to Develop Trustworthy Systems)*, *24*(7), 815-835.

Najork, M. (2009). *Web crawler architecture.*

Ng, A. (2021). *Big data to good data: Andrew ng urges ml community to be more data-centric and less model-centric.* Retrieved from `https://analyticsindiamag.com/big-data-to-good-data-andrew-ng-urges-ml-community-to-be-more-data-centric-and-less-model-centric/`

Noy, N., Brickley, D., & Burgess, M. (2019). Google dataset search: Building a search engine for datasets in an open web ecosystem. In *The world wide web conference* (pp. 1365–1375).

NRC. (2013). *Frontiers in massive data analysis.* National Research Council. Washington, DC: The National Academies Press. https://doi.org/10.17226/18374.

Okoli, C., & Pawlowski, S. D. (2004). The delphi method as a research tool:an example, design considerations and applications. *Informaton & Management*, *42*(1), 15–29.

Ottenheijm, S. (2017). *Big data in healthcare: Definition, application, and challenges* (Tech. Rep.). Nictiz.

Paleyes, A., Urma, R.-G., & Lawrence, N. D. (2022). Challenges in deploying machine learning: A survey of case studies. *ACM Comput. Surv.*.

Pawar, A., & Mago, V. (2018). Calculating the similarity between words and sentences using a lexical database and corpus statistics. *arXiv preprint arXiv:1802.05667*.

Pipino, L., Lee, Y., & Wang, R. (2002). Data quality assessment. *Communications of the ACM*, *45*(4), 211–218.

Poelmans, J., Ignatov, D. I., Kuznetsov, S. O., & Dedene, G. (2013). Formal concept analysis in knowledge processing: A survey on applications. *Expert Systems with Applications*, *40*, 6538–6560.

Polyzotis, N., Roy, S., Whang, S. E., & Zinkevich, M. (2018). Data lifecycle challenges in production machine learning: A survey. *SIGMOD Record*, *47*(2), 17–28.

Porello, D., & Endriss, U. (2011). Ontology merging as social choice. In *Computational logic in multi-agent systems - lecture notes in computer science* (pp. 27–56). Springer, Berlin.

Rekatsinas, T., Dong, X., Getoor, L., & Srivastava, D. (2015). Finding quality in quantity: The challenge of discovering valuable sources for integration. In *Cidr*.

Rekatsinas, T., Dong, X., & Srivastava, D. (2014). Characterizing and selecting fresh data sources. In *Proceedings of the 2014 acm sigmod international conference on management of data* (pp. 919–930).

Roh, Y., Heo, G., & Whang, S. E. (2021). A survey on data collection for machine learning: A big data - ai integration perspective. *IEEE Transactions on Knowledge and Data Engineering*, *33*(4), 1328-1347. doi: 10.1109/TKDE.2019.2946162

Sansone, S.-A., Gonzalez-Beltran, A., Rocca-Serra, P., Alter, G., Grethe, J. S., Xu, H., ... others (2017). Dats, the data tag suite to enable discoverability of datasets. *Scientific data*, *4*(1), 1–8.

Schema.org, dataset. (n.d.). Retrieved from `https://schema.org/Dataset` (Accessed on 2022-09-11)

Seng, J. J. B., Monteiro, A. Y., Kwan, Y. H., Zainudin, S. B., Tan, C. S., Thumboo, J., & Low, L. L. (2021). Population segmentation of type 2 diabetes mellitus patients and its clinical applications-a scoping review. *BMC medical research methodology*, *21*(1), 1–19.

Shull, F., Seaman, C., & Zelkowltz, M. (2006). Victor r. basili's contributions to software quality. *IEEE Software*, *23*(1), 16-18. doi: 10.1109/MS.2006.33

Solingen, R. V., & Berghout, E. (1999). *The goal/question/metric method: a practical* *guide for quality improvement of software development.* THE McGRAW-HILL COMPANIES, London.

Solingen, V., Basili, V., Caldiera, G., & Rombach, H. (2002). Goal question metric (gqm) approach. In *Encyclopedia of software engineering.* Wiley Online Library.

Sordo, M., Tokachichu, P., Vitale, C. J., Maviglia, S. M., & Rocha, R. A. (2017). Modeling contextual knowledge for clinical decision support. In *Amia annual symposium proceedings (published 2018)* (p. 1617-1624).

Standard: Metadata. (n.d.). Retrieved from `https://datasf.org/resources/metadata-standard` (Accessed on 2022-10-15)

Strozyna, M., Elden, G., Filipiak, W. A. D., Malyszko, J., & Wecel, K. (2018). A framework for the quality-based selection and retrieval of open data - a use case from the maritime domain. *Electron Markets*, *28*, 219–233.

Stumme, G., & Maedche, A. (2001). Fca-merge: Bottom-up merging of ontologies. In *Proc. 17th intl. conf. on artificial intelligence (ijcai '01), seattle, wa, usa* (pp. 225–230).

Takeuchi, H., & Yamamoto, S. (2020). Business analysis method for constructing business–ai alignment model. *Procedia Computer Science*, *176*, 1312-1321. Retrieved from `https://www.sciencedirect.com/science/article/pii/S1877050920320408` (Knowledge-Based and Intelligent Information & Engineering Systems: Proceedings of the 24th International Conference KES2020) doi: https://doi.org/10.1016/j.procs.2020.09.140

Trojahn, C., Vieira, R., Schmidt, D., Pease, A., & Guizzardi, G. (2021). Foundational ontologies meet ontology matching: A survey. *Semantic Web*(Preprint), 1–20.

Valtchev, P., Hacene, M. R., & Missaoui, R. (2003). A generic scheme for the design of efficient on-line algorithms for lattices. In *International conference on conceptual structures* (pp. 282–295).

Valtchev, P., Missaoui, R., & Lebrun, P. (2002). A partition-based approach towards constructing galois (concept) lattices. *Discrete Mathematics*, *256*(3), 801-829. Retrieved from `https://www.sciencedirect.com/science/article/pii/S0012365X02003497` (LaCIM 2000 Conference on Combinatorics, Computer Science and Appl ications) doi: https://doi.org/10.1016/S0012-365X(02)00349-7

van Rossum, G. (2018). *Python tutorial.* Python Software Foundation.

Wan, K. (2006). *Lucx: Lucid enriched with context* (Unpublished doctoral dissertation). Concordia University.

Wan, K., Alagar, V., & Pacquet, J. (2005). An architecture for developing context-aware systems. In *Proceedings of 2nd international workshop on modeling and retrieval of context (marc2005)* (p. 48-62). LNCS, Springer-Verlag, Vol. 3946.

Weiskopf, N. G., Hripcsak, G., Swaminathan, S., & Weng, C. (2013). Defining and measuring completeness of electronic health records for secondary use. *J. Biomedical Inform*, *46*(5), 830–836.

Wetherill, D. (2016). *Broken links: Why analytics investments have yet to pay off* (Tech. Rep.). ZS Associates: www.zsassociates.com.

Wille, R. (2009). Restructuring lattice theory: an approach based on hierarchies of concepts. In *International conference on formal concept analysis* (pp. 314–339).

Wolff, K. E. (1988). *Einführung in die formale begriffsanalyse.*

Zeil, S. J. (2020). *The algebra of bog o.* https://www.cs.odu.edu/ zeil/cs361/latest/Public/algebra/index.html.

Zhang, M., Hadjieleftheriou, M., & et al. (2011). Automatic discovery of attributes in relational databases. In *Proceedings of sigmod'11* (pp. 1–12).

Zhang, P., Wang, F., Hu, J., & Sorrentino, R. (2014). Towards personalized medicine: Leveraging patient similarity and drug similarity analytics. In *Proceedings of amia joint summits transci. sci.* (pp. 132–136).

Zhang, S., Hu, Y., & Bian, G. (2017). Research on string similarity algorithm based on levenshtein distance. In *2017 ieee 2nd advanced iinformation technology, electronic and automation control conference* (pp. 2247—2251).