2015

Graphical Database Architecture For Clinical Trials

Aiman G. Baset

North Carolina Agricultural and Technical State University

Graphical Database Architecture for Clinical Trials

Aiman G. Baset

North Carolina A&T State University

A thesis submitted to the graduate faculty

in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Department: Computer Systems Technology

Major: Information Technology

Major Professor: Dr. Naser El-Bathy

Greensboro, North Carolina

2015

.

The Graduate School

North Carolina Agricultural and Technical State University

This is to certify that the Master’s Thesis of

Aiman G Baset

has met the thesis requirements of

North Carolina Agricultural and Technical State University

Greensboro, North Carolina

2015

Approved by:

Dr. Naser El-Bathy
Major Professor

Dr. Clay Gloster
Committee Member

Dr. Rajeev Agrawal
Committee Member

Evelyn Sowells
Committee Member

Dr. Clay Gloster
Department Chair

Dr. Sanjiv Sarin
Dean, The Graduate School

Biographical Sketch

Aiman Baset, a senior year Masters student at North Carolina A&T School of Technology, has a commitment and appreciation for technology learning and applying. In 2010, Mr. Baset graduated from the University of Kalamoon, Syria where he earned a BA in business and administration with a minor in management of information systems (MIS). During his an undergraduate program, he worked as a business journalist for Syria Report website. He design and developed a corporate database for all Syrian businesses. Also he worked for Management Advisory Services and Consultancy (MASC) as an Internal Auditor. In addition, he served as a translator for Certification of Internal Auditor (CIA) training materials and slides. Currently Aiman serves as a technology intern with VF's Jeanswear.

Dedication

For my country Syria, for my father, Ghassan, my mother Hasibah, Bashira, Mo, Huda, Mazen and Baset family.

Acknowledgements

I am deeply grateful and indebted to my advisor Dr. Naser El-Bathy, Assistant Professor at the Department of Computer Systems Technology, for introducing me to the world of health informatics, for keeping my attention to the state-of-the-art research directions, for all his patience, invaluable advice and continuous support and encouragement during my graduate studies.

I would like to thank my committee members, Dr. Clay Gloster, a Professor and the Chair of the Department of Computer Systems Technology and Interim Associate Dean of School of Technology, from whom I learned a lot, Dr. Rajeev Agrawal, Assistant Professor at the Department of Computer Systems Technology, and Dr. Evelyn Sowells, Assistant Professor at the Department of Computer Systems Technology, for serving on my thesis committee and for providing me with useful comments and valuable suggestions.

Finally, I thank my parents, Ghassan and Hasibah Baset, and my siblings who planted the seeds of this journey early in my life. Also, I thank the Hamoush's for their extraordinary support while I pursued my educational dream.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

AS – Application Server

API – Application Program Interface

CT – Clinical Trials

DB – Database

DM – Data Mining

DME – Data Mining Engine

DW – Data warehouse

ETL – Extraction, Transformation, Loading

GDA – Graphical Database Architecture

GUI – Graphical User Interface

HCI – Human Computer Interaction

HI - Health Informatics

HTTP - Hypertext Transfer Protocol

IA – Information Architecture

IE – Information Extraction

IR – Information Retrieval

IS – Information System

KDD – Knowledge Discovery in Databases

Neo4j – Neo Technology for Java

NoSQL – Not Only Structured Query Language

OLTP – On-Line Transaction Processing

REST - Representational State Transfer

SOA – Service-Oriented Architecture

SQL – Structured Query Language

UI – User Interface

WWW – World Wide Web

XML – Extensible Markup Language

Abstract

The general area of the research is Health Informatics. The research focuses on creating an innovative and novel solution to manage and analyze clinical trials data. It constructs a Graphical Database Architecture (GDA) for Clinical Trials (CT) using New Technology for Java (Neo4j) as a robust, a scalable and a high-performance database.

The purpose of the research project is to develop concepts and techniques based on architecture to accelerate the processing time of clinical data navigation at lower cost. The research design uses a positivist approach to empirical research.

The research is significant because it proposes a new approach of clinical trials through graph theory and designs a responsive structure of clinical data that can be deployed across all the health informatics landscape. It uniquely contributes to scholarly literature of the phenomena of Not only SQL (NoSQL) graph databases, mainly Neo4j in CT, for future research of clinical informatics. A prototype is created and examined to validate the concepts, taking advantage of Neo4j's high availability, scalability, and powerful graph query language (Cypher).

This research study finds that integration of search methodologies and information retrieval with the graphical database provides a solid starting point to manage, query, and analyze the clinical trials data, furthermore the design and the development of a prototype demonstrate the conceptual model of this study. Likewise the proposed clinical trials ontology (CTO) incorporates all data elements of a standard clinical study which facilitate a heuristic overview of treatments, interventions, and outcome results of these studies.

CHAPTER 1

INTRODUCTION

The Modernization of the medical and healthcare industries continues to experience numerous challenges despite its major successes in the past two decades. Development of concepts and techniques to help solve the healthcare problems is the focal point of such challenges. The results of these problems varies from a problem to another. The causes for these challenges include several factors. The first factor is the absence of open software systems' integrity. The second factor is an inefficient medical and healthcare information retrieval processes. The third factor is redundant and inaccurate data. The fourth factor is IT inability to align with the business strategy. Finally, poor quality of information results in lack of information that analysts require to plan strategy for the medical and healthcare industries.

This research encompasses medical and healthcare organizations; that include its domains, its employed technology, product scope, customer orientation, and markets (Creswell, 2003). This study focuses on an innovative solution to aggregate, manage, mine, and analyze clinical trials data and improve compliance with the original Health Information Technology Standards (Bowen, 2011). The research develops a Graphical Database Architecture (GDA) for Clinical Trials (CT). It captures the challenges of clinical trials data to develop concepts and techniques based on the architecture to help solve the problems of data management. Our long-term goal is to define and implement an innovative intelligent open source software system to decrease the problems associate with clinical trials data in healthcare using cultural algorithm such as swarm algorithm. Figure 1 describes paradigm of the graphical database architecture that consist of three layers, user layer which where the user inter a search term or a cypher query, the graphical database clinical trial layer, and the data sources layer. This layer retrieve the data from the clinical trials site or the database management component.

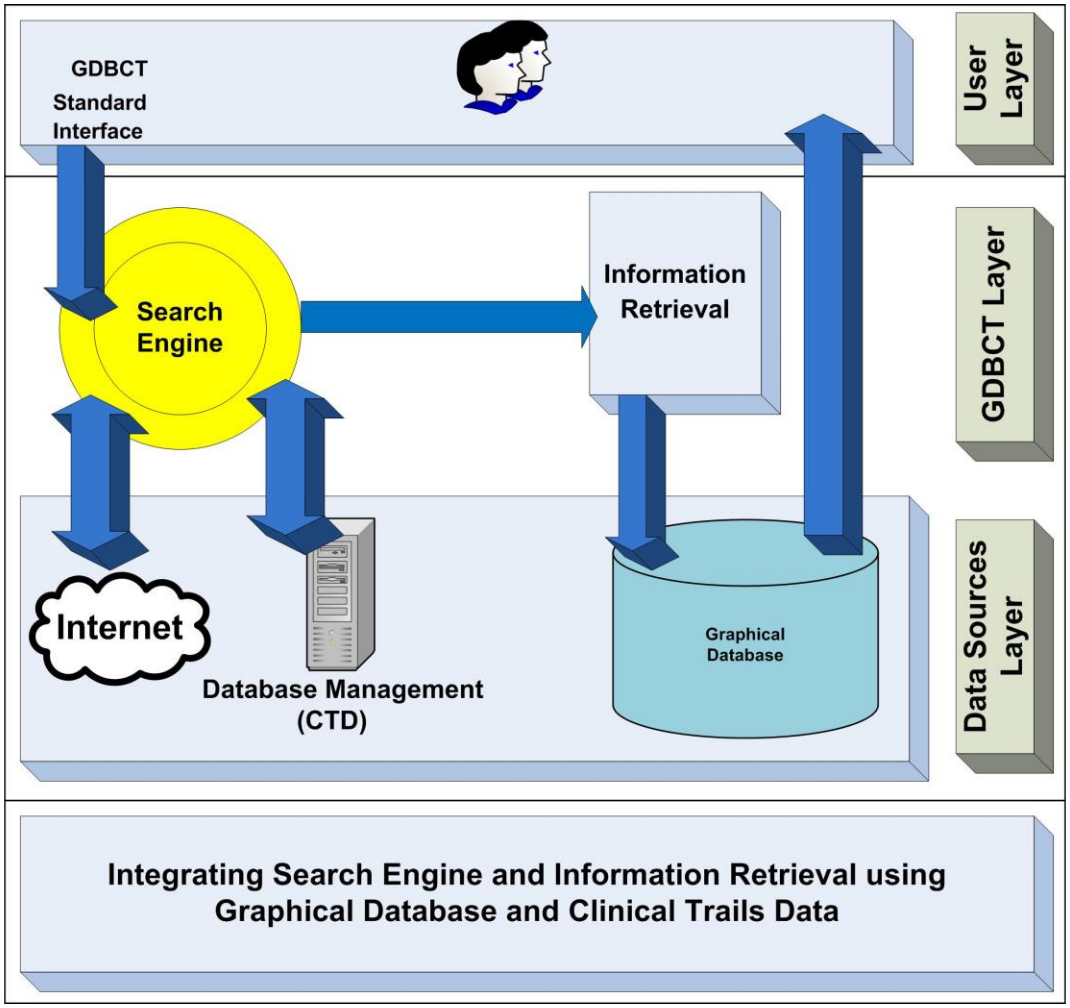

GDBCT
Standard
Interface
User Layer
Search Engine
Information Retrieval
GDBCT Layer
Internet
Database Management (CTD)
Graphical Database
Data Sources Layer
Integrating Search Engine and Information Retrieval using Graphical Database and Clinical Trails Data

Figure 1. The Architecture of Graphical Database for Clinical Trails

This study is a new attempt to apply GDA principles by providing dynamic services that have concrete meaning on the medical and healthcare industries level to reduce costs, resources, time, overhead, and minimize health risks. The implementation of GDA has acceptable benefits such as agility, integrity, and a long-term strategy. This strategy increases flexibility of the healthcare IT infrastructure (El-Bathy & Gloster & Azar, 2012). New medical and healthcare requirements will be developed in the short term via the reusability of existing business logic and data models (Watson, 2006).

The development of an automated Graphical Database Architecture for Clinical Trials system is regarded as a major and a significant component for clinical trials data. Researchers have often studied general technical types of information systems which cannot entirely solve the problems of Medical and Healthcare information. Therefore, the reality is that the industries' organizations still face severe obstacles mainly in managing clinical trials information that have adapted over time based on doctors' requests. Our system will allow medical personnel to query a database to determine if similar clinical trials have previously been conducted

The required elements for choosing the area of the study for this research have been established based on extraordinary experience in Information Technology, Healthcare organizations, and an Academic Degree. The problem has emerged as management of clinical trials data that healthcare professionals encounter. The reasons for such problem are inefficient Medical and Healthcare information retrieval processes due to numerous grounds. These primarily include huge redundant data, data inaccuracy, IT inability to align with the business strategy, and the absence of software systems' integrity (Watson, 2006).

Thus, efficiency, performance, and quality are essential to eliminate not only the effects of the problem but also its causes. Hence, the long-term goal is to define and implement an innovative Graphical Database Architecture (GDA) for Clinical Trials (CT) to improve the accuracy of Clinical Trials data through accelerated information retrieval. However, determining the requirements for accomplishing the long-term goal is the main part for successfully achieving the specified objectives. The study for this research establishes a prototype model that supports the alignment between IT goals and Medical and Healthcare organizations strategy. The prototype model defines a target technology environment that supports the organization strategic plan. It allows an organization to meet its strategic, tactical, and operational needs, and updates IT processes. Not only is the performance of an organization improved by applying the methodology of strategic alignment, but also the organization’s effectiveness (Gutierrez & Serrano, 2008). A successful IT strategic plan is based on four main priorities. These are IT-business alignment escalation, IT governance enhancement, IT significance improvement, and IT investments grade. The IT strategic plan should follow a timetable via a roadmap and it is exposed to alteration based on the business process changes (Cullen & Cecere & Symons & Cameron & Cardin & Orlov & Belanger, 2007).

The development of IT strategic plan consists of five phases. These phases include the purpose of the plan, the business requirement, and the business requirements’ validation. The phases also include the procedures needed for eliminating the gaps between the state of “to be” and the IT processes state, and the plan approval (Cullen & Cecere & Symons & Cameron & Cardin & Orlov & Belanger, 2007).

In this research, the concept of big Data is poised to deliver revenues efficiently for enterprises such that innovation of healthcare information technology (IT) will change fast in few decades (Chambers; Dhiraj, & Minelli, 2013). According to International Data Corporation in 2012, Figure 2 depicts the growth of data from 2005 to 2017.

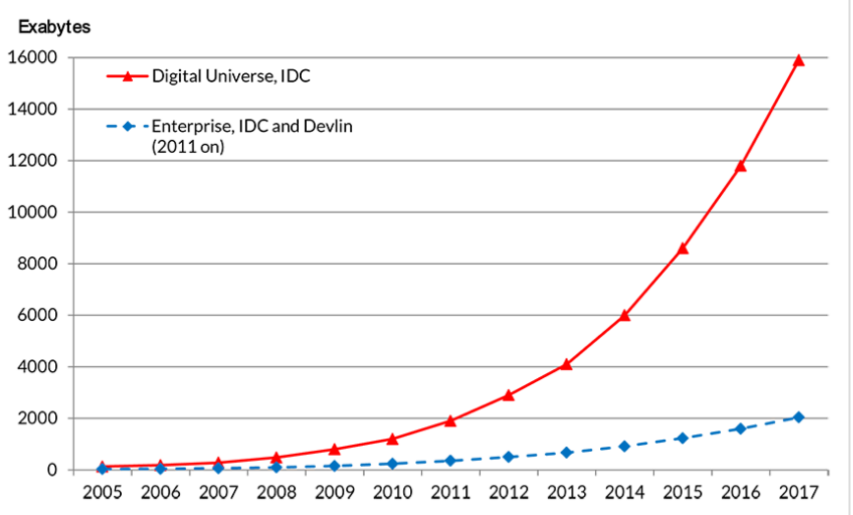


Figure 2. Data Size explosion

Figure 3 shows a timeline of recent technology developments (Chambers; Dhiraj, & Minelli, 2013). Back in the 1980s enterprise resource planning systems (ERP) (i.e. SAP) were the central component of any corporation information system ecosystem, then more sophisticated and specialized systems emerged like customer relationship management (CRM) that manage a company's collaborations with customers, the new millennium was associated with the dot com boom or internet boom, and now the narrative is to consider methods and concepts to manage the data explosion using machine learning systems, data mining, and big data solutions.

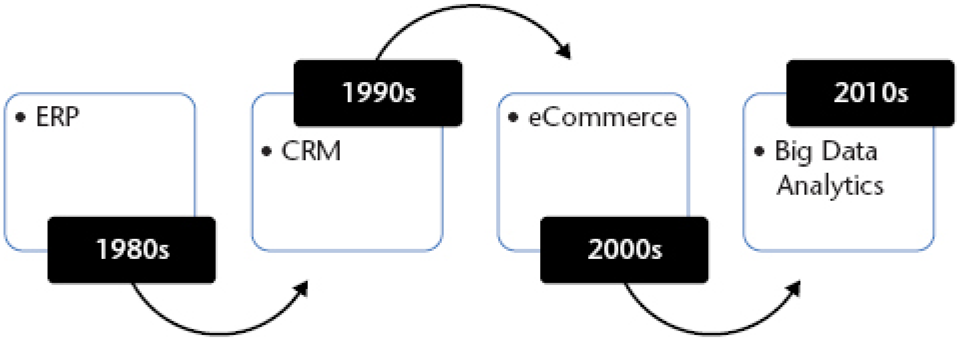


Figure 3. Timeline of Recent Technology Developments

In clinical trials, data accessibility is required to be provided at user-level based on user's queries and queries' results analysis. The users are insurance companies, pharmaceutical companies, professional medical staff, doctors, schools, and research centers. Further the clinical trials data provide a valuable source of information that can be useful for all medical professionals to gain insights on the conducted interventional and observational clinical studies

One of the technologies used to aggregate, manage, mine, and analyze big data is non-relational databases because the database does not store data in traditional tables and link tables to each other through relationships (Kumar Singh, 2011). In the same direction NOSQL an acronym for Not Only Structured Query Language intended for managing data in non-relational databases. But still have the ability to create, insert, query, update, and delete data in a schema-less landscape (Vaish, 2013).

1.1 Problem Statement and Research Questions

The problem of this research is identified as unnecessary treatments and testing for diseases. This problem is a result of unproductive medical and healthcare information retrieval processes. This research addresses the need for software systems that allow medical and healthcare organizations to meet its strategic, tactical and operational goals faster at lower cost (El-Bathy & Gloster & Azar, 2012).

One specific research question which arises is: How can the integration of search methodologies, information retrieval, and graphical database be used to determine the best treatment for a certain disease using clinical trials data?

Querying a database requires that the user has developed a deep knowledge of database concepts. This requirement may include the following: concepts of table fields and tuples, primary and foreign keys, entities and relationships (Guimarães da Silva, 2014). It also requires that the user masters a database query language such as SQL. Additionally, querying requires that the user knows what kind of data is stored in the database, in order to define what to query. Given these restrictions, the user's task of querying a database requires a cognitive overload which may not be overlooked, in part because typical users do not have such theoretical knowledge and do not know the internal organization of the database to be queried (Guimarães da Silva, 2014).

When analyzing a query's results provided by a Relational Database Management System (RDBMS), it is difficult to get a data overview and to detect patterns, and trends in the data. Graphical and interactive representations of data, as proposed by the Information Visualization area (Card et al., 1999) are appropriate for this kind of analysis. They provide distinct levels of data overview, details on demand, and interactive capabilities of data reorganization and filtering, among other useful resources.

Interactive filtering techniques such as dynamic queries provide user controls for selecting relevant data and for querying details, providing fast answers without the need to learn command-line querying syntax. When applied together, filtering and other techniques enhance users' formulation of an internal model about the data analysis (Shneiderman, 1994).

This research study proposes the following steps to create a Graphical Database Architecture (GDA) for Clinical Trials data for data mining, data management, data visualization, and data accessibility capabilities:

1. Collect clinical trials data in Extensible Markup Language (XML) format
2. Wrangling, cleaning, and preparing data (Convert xml to Neo4j format)
3. Create a graphical database model in Neo4j
4. Generate nodes and edges for the Neo4j graphical database.

5. Load the converted clinical trials data into the created Neo4j graphical database by executing Cypher statements

1.2 Purpose of Research

The purpose of this research is to use a new type of database, a database that uses graph structures with nodes, edges and properties, to represent and store information. The GDA tests the efficiency and effectiveness compared to the relational database management systems. This requires building and designing a graphical database to extract and process clinical trial data to be analyzed, published, and shared. A key feature of graph databases is they offer a unique vantage point of data, where the solution to a problem is seen as a conceptually defined traversals through its vertices and edges.

Clinical trials are used to evaluate what treatment works best against a certain disease based on data acquisition, management, retrieval, and analysis. Clinical trials have to be planned, designed, and tested.

This research study proposes a solution that provides data management capabilities, data collection, data representation, data navigation, and mining using the query language of graph databases specifically Neo4J's language Cypher. The study uses the solution as a database (DB) management tool and a learning management system.

Though there are several excellent general reviews of NoSQL databases and technologies each to some extent reflects the authors' personal research interests and expertise. Due to the pace of development and breadth of research, a truly comprehensive review is beyond the scope of this thesis.

1.3 Significance of the Research

A study of analyzing clinical trials data to identify more indications and discover adverse effects. The study to analyze, near real-time collections of adverse case reports enables identifying events that were hinted at in the clinical trials but that did not have sufficient statistical significance.

So this research study proposes a new concept. It defines a new mechanism for data management. By using graph databases of NoSQL, faster traversing is accomplished. Also, joining or merging relationships is very fast and efficient. The relationship between nodes is not calculated at query time but persists as a relationship. Traversing persisted relationships is faster than calculating them for every query.

Semi-structured data models are oriented to model semi-structured data (Buneman & Abiteboul, 1997). They deal with data whose structure is irregular, implicit, and partial, and with schema contained in the data. One of these models is OEM (Object Exchange Model) (Papakonstantinou, Garcia-Molina & Widom, 1995). It aims to express data in a standard way to solve the information exchange problem. This model is based on objects that have unique identifiers, and property value that can be simple types or references to objects. Another data model is the XML data model (Bray, Paoli, Sperberg-McQueen, Maler, & Yergeau, 2006). XML has an ordered-tree like structure and the information about data in XML is part of the data set. Semi-structured documents, such as XML, can be effectively modeled using a rooted ordered labeled tree or graph. A graph consists of a set of nodes (or vertices) that are connected by edges.

Each edge has two nodes associated with it. A path is defined as a finite sequence of edges that connect nodes. A rooted tree has its top-most node defined as the root that has no incoming edges and there is a single unique path between any two nodes. A node *u* is said to be a parent of node *v*, if there is a directed edge from *u* to *v*. Node *v* is then said to be a child of node *u*. Nodes with the same parent are called siblings. The ancestors of a node *u* are the nodes on the path between the root and *u*, excluding *u* itself. The descendants of a node *v* can then be defined as those nodes that have *v* as their ancestor.

1.4 Contribution to Knowledge

This research study is a new hypothetical concept defining the requirements for a graphical database model for clinical trials data. It integrates data mining capabilities, visualization representation, and better performance to help many stakeholders who are involved in one or another in the clinical trials field like insurance companies, pharmaceutical companies, professional medical staff, doctors, schools, and research center.

This study adds to the field of health informatics by applying an innovative model and a new way of handling semi-structured data. The new concept try to introduce a dynamic graphical database model that provides efficiency, high availability, handling a huge number of nodes (32 billion), edges (32 billion), and up to 64 billion properties. That will allow all stakeholders involved or interested in clinical trials data to utilize it for better data management and faster processing time. Although giant steps have been made in recent years in the field of health informatics, there remains an open question as to improve, find new way of to answer research questions.

CHAPTER 2

LITERATURE REVIEW

This review is intended to summarize current knowledge about Medical and Healthcare organizations. It is a crucial mechanism for the development of a new theory to accelerate processing time, and provide a new direction for future research and management of the alignment of IT goals with Medical and Healthcare organizations strategy.

In the literature analysis, alternates are used to differentiate between the characteristics of the information that had been saved in files on a computer as a soft copy and the information that had been saved in less thoroughly reviewed outlet.

This chapter identifies more focused literature requirements for the literature survey phase of the research. More specific topics give a solid background and knowledge to develop a model and methodology to solve the research problems.

This research study builds the prototype model. Building stage is the phase of the requirements, design, specifying software tools, and designing the interfaces. Prototyping stage is the phase of coding and testing.

The purpose of this study is to verify that the model works. The model provides prototyping stage developer with outlines to follow. There are three approaches. The first is top down. That is, the research starts from the top component and ends with the bottom component. The second is bottom up; the research starts from the bottom component and ends with the top component. The third is hybrid; the research starts from the middle of the model and ends with both the top and bottom components.

2.1 Health Informatics

Health informatics is a developing field that is becoming progressively important to the quality and effective distribution of healthcare. Now any modern health organization can't function efficiently without an electronic database storing, processing, and communicating patient-related data. With the increasing complexity of chronic illness and the number of medications prescribed in an elderly population, information technologies and decision support tools can be used to help to prevent medication errors, improve the analyzing capabilities of the health and medical data (Lui, 2010).

As an academic discipline evolves, one naturally asks what makes it different from existing fields and how do the practitioners of the discipline stand out from current workers. This chapter of the thesis presents the link between health informatics (HI) especially Clinical trials data and the graph databases, the development of health informatics (HI) as a discipline and how this relates to the people who work in it will be investigated.

Health informatics (HI) is the scientific discipline concerned with the creation, storage, processing, transmission, and use of health-related data, information and knowledge (Shortliffe & Cimino, 2006, p. 24).

It is a broad field and its workers have a wide range of responsibilities and tasks. Part of the reason for the breadth is the natural evolution of health informatics (HI) as a discipline. The origin of the formal discipline was strongly influenced by the development of the electronic computer. Indeed, the term informatics came from European developments in computer science. Informatics was probably first created by the father of German computer science, Karl Steinbuch, in his 1957 paper, "Informatik: automatische informationsverbeitung", which translates to "Informatics: automatic information processing" (Widrow, Hartenstein, & Hecht-

Nielsen, 2005). The french computer science pioneer, Philippe Dreyfus, used the term in establishing the Société d'Informatique Appliqueé in 1962 (Fourman, 2002, p. 1)

Informatics became more to do with information rather than simply computers. However, readers should be wary; there are many who consider informatics a field of computer science, or indeed as a synonym (Buerck & Feig, 2006; Kane, Brewer, Goldman, & Moidu, 2006; Smith & Buerck, 2007).

Whether information exists in print or in bits and bytes, health knowledge cannot be used if it cannot be accessed. Psychology, and in particular, cognitive science, indicates how health practitioners think and work and how computer systems might best be designed to support them e.g. understanding the cognitive workings of health workers that lead to the failure of a neonatal intensive care decision-support system (Alberdi, et al., 2000).

Health informatics (HI) supports all processes including administration, governance, clinical care and research. Since the informational problems encountered in certain areas can differ in content, subareas of health also have their own informatics knowledge; one can recognize subfields such as medical informatics, nursing informatics, public health informatics, patient (consumer) informatics and pharmacy informatics. As for computing, it has been said that the computer scientist is a tool smith, and his success is determined by whether people use his tools (Brooks, 1996).

Healthcare tracks a specific biomedical scientific paradigm, where there is an obligation on understanding the scientific principles behind healthcare activities. This differs from the engineering perspective to which IT professionals are accustomed (Tichy, 1998; Zannier, Melnik, & Maurer, 2006). Drug therapy is a good example; health professionals require scientific proof that a drug is beneficial, often in the form of controlled experiments (clinical trials).

Clinical informaticians: This is a role that extends the idea of the ad hoc health IT worker from early history; these people were health workers who picked up IT as it became incorporated into health organizations. Today, a formal place is recognized for those who hold dual roles as clinicians (doctors, nurses, etc.) and health informaticians. Health Informatics is, and should be, driven by health informational problems. The best exposure to such informational problems is active clinical practice. These people are seen by general clinicians as understanding the nature of health service delivery and as a referral point for informational problems. Indeed, clinical informaticians can make up part of a clinical team. Just as the ad hoc pioneers became beacons for health informatics (HI), they too become a visible resource for others and are the modern day champions. Recently, the American Medical Informatics Association began work on formalizing clinical informatics as a specialist area for physicians; these individuals combine their knowledge of patient care and informatics methods and tools to assess the knowledge needs of health workers and patients, and to develop and evaluate clinical computer systems (Safran et al., 2009).

Health Informatics is the scientific discipline concerned with how best to use data, information and knowledge in medicine to improve patient care. Information science principles and computer and communications technology are the tools that Health Informatics professionals use. The field is still evolving but one can demonstrate that professionalism exists. There is a body of scientific theory, set of skills, knowledge and competencies, and professional attitudes shared by practitioners, despite a range of informatics roles.

The need for Health Informatics specialists is clear. The domain expert-IT paradigm is insufficient for non-trivial health computing. The Health Informatics professional brings her knowledge of health and a wider kit of tools and understanding to apply to informational problems.

The Health Informatics professional is likely to become increasingly important to healthcare organizations, clinicians and patients as the cost and complexity of healthcare increase and as technologies advance and become more useful. Education will continue to play an important role in the qualification of Health Informatics professionals, despite the various standards and curricula present today. Similarly, certification/accreditation and regulatory bodies will become increasingly relevant; a number of professional associations currently exist and matters of jurisdiction will have to be addressed.

2.2 Clinical Trials

Although clinical trials are a substantial and rapidly growing international business, they are not well understood by the general public. They are often seen as obscure and are sometimes surrounded by controversy about the ethical aspects that touch the nature of the early side affects.

Clinical trials are a systematic and rigorous procedure to distinguish successful therapies and medicines from unsuccessful ones. Gradually, more structured experimentation began to take place. Experiments of the British naval doctor James Lind in the mid-eighteenth century were designed to discover the causes of scurvy, a disease that had decimated British naval crews.

The experiments relied on comparative groups of men, giving one group vinegar and the other group citrus fruits to treat scurvy. As a result of these experiments, which demonstrated the efficacy of oranges and lemons against scurvy, the Royal Navy began supplying citrus fruits to its ships in 1795. However, it took more than a hundred years from those early experiments for the modern system of drug testing to be born.

While in the United States, not until 1938 did the U.S. Food Drug and Cosmetic Act subject new drugs to premarket safety evaluation and require Food and Drug Administration (FDA) regulators to review both preclinical and clinical test results. However, even this law did not precisely specify the kinds of tests required for approval, so it gave regulators only limited powers for negotiation with the pharmaceutical industry. Public health disasters caused by poorly tested drugs, such as thalidomide, showed that the system was not adequate to the task. The thalidomide disaster and other, similar ones ultimately prompted the passing of the 1962 Drug Amendments. Those laws explicitly stated that new drug approvals would rely on substantial scientific evidence of a drug's safety and efficacy. Therefore, the FDA was obliged to base its approvals on "adequate and well-controlled studies." The modern clinical trial, based on statistically controlled, evidence-based results, was born and eventually became the requirement for new drug approvals by the FDA. The FDA approval procedure is regarded today as a world standard of the drug industry.

Preclinical studies involve two types of experiments: test tube ones, known as **"in vitro,"** and also experiments carried out with animals or cell cultures, known as "**in vivo**." The experiments apply different doses of the drug under study to develop acute toxicity and toxicity ranges and to establish a pharmacological profile of the drug. The tests' results inform the decisions related to granting the drug candidate with an investigational new drug (IND) status and open the way for further testing of the drug candidate under the actual clinical trials regime.

Clinical trial (also called an interventional study), the participants receive specific interventions according to the research plan or protocol created by the investigators. These interventions may be medical drugs or medical devices; medical procedures; or applying changes to participants' behavior. When a new product or approach is being studied, it is not usually known whether it will be helpful, harmful, or no different than available alternatives so the investigators work to determine the safety and efficiency of the proposed intervention by measuring specific outcomes on the participants.

Generally clinical trials used in drug development are described or divided by phase. These phases (will be discussed later) are defined by the Food and Drug Administration (FDA).

A principal investigator usually led the clinical trial, a research team consists of doctors, physicians, nurses, and health care professionals' works under the supervision of the investigator. These trials take place in different locations hospitals, research centers, medical schools, and a community of health professionals. ("Learn About Clinical Studies." *Home*. N.p., n.d. Web. August. 2012).

Clinical trials are planned to add to medical knowledge related to the treatment, diagnosis, and prevention of diseases. To be precise:

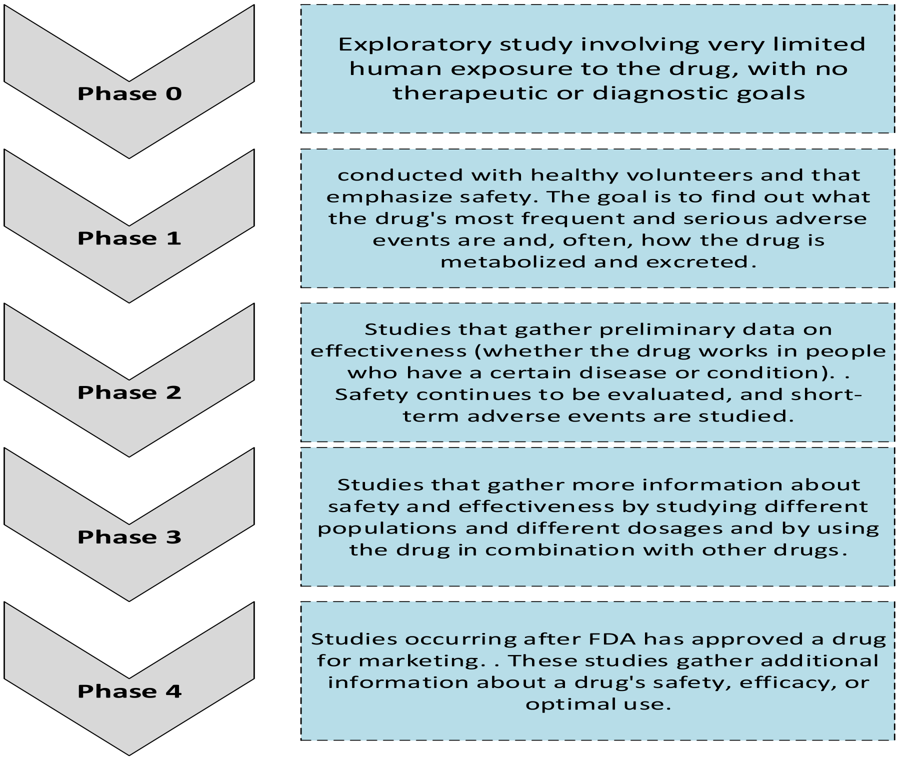
- Evaluating interventions for treating a disease, syndrome, or condition
- Finding ways to prevent the initial development or recurrence of a disease or condition.
- Examining methods for identifying a condition or risk factors for that condition
- Exploring ways to measure and improve the comfort and quality of life of people with a chronic illness.

Figure 4 illustrates that actual clinical trials of a new drug typically proceed in four phases. **Phase I** trials constitute the first stage of testing in human subjects. Usually, a small group of 20–100 healthy volunteers are selected. **Phase I** includes trials designed to assess the safety profile and initial tolerability, pharmacokinetics, and pharmacodynamics of a drug candidate.

Phase II is performed on larger groups (100–300) of patients who have the disease the drug aims to treat or prevent, to determine preliminary effectiveness. This outcome is designated as confidence in mechanism (COM). Safety assessments that started in phase I are continued, but in a larger group of volunteers and patients to assess common short-term side effects and risks of the drug candidate. These safety tests should result in reaching a milestone called confidence in safety (CIS). Phase II studies are sometimes divided into phase IIA (related to frequency of dosing) and phase IIB (efficacy at prescribed dosage).

Phase III trials expand upon the preliminary data for effectiveness and monitor longer-term safety to determine the overall benefit–risk profile in larger groups of patients. The results form the basis that informs the labeling information, which determines marketing elements of the new drug. Phase III is costly and usually constitutes the “pivotal trials” regulators weigh strongly in their approval or disapproval decision

Phase IV in some scenarios, some clinical trial studies may need to be conducted as well as the larger ‘pivotal’ studies. However, if these studies are small may not be necessary before submitting for licensing approval but are conducted after approval and are called phase IV (O'Neill & Hopkins, 2011)



Phase 0

Exploratory study involving very limited human exposure to the drug, with no therapeutic or diagnostic goals

Phase 1

conducted with healthy volunteers and that emphasize safety. The goal is to find out what the drug's most frequent and serious adverse events are and, often, how the drug is metabolized and excreted.

Phase 2

Studies that gather preliminary data on effectiveness (whether the drug works in people who have a certain disease or condition). . Safety continues to be evaluated, and short-term adverse events are studied.

Phase 3

Studies that gather more information about safety and effectiveness by studying different populations and different dosages and by using the drug in combination with other drugs.

Phase 4

Studies occurring after FDA has approved a drug for marketing. . These studies gather additional information about a drug's safety, efficacy, or optimal use.

https://clinicaltrials.gov/ct2/help/glossary/phase

Figure 4. Clinical Trials Phases

2.3 NOSQL Databases

For over decades, relational databases have been used to store structured data. The data is sub-divided into groups, referred to as tables (Vaish, 2013). The tables store well-defined units of data in terms of type, size, and other constraints. Each unit of data is known as a column while each unit of the group is known as a row. The columns may have relationships defined across themselves, for example parent-child, and hence the name relational databases. And because consistency is one of the critical factors, scaling horizontally is a challenging task.

2.3.1 NOSQL. With the growth of big web applications, research has poured into handling data at scale. One of the outputs of these studies is non-relational database, in general referred to as NoSQL database. One of the main problems that a NoSQL database solves is scale, among others (Vaish, 2013).

Clearly NoSQL databases don't use SQL. But most of them do have query languages, and it makes sense for them to be similar to SQL in order to make them easier to learn.
An important characteristic of these databases is that they are generally open-source projects. Although the term NoSQL is frequently applied to closed-source systems, there's a notion that NoSQL is an open-source phenomenon (Fowler & Sadalage, 2012).

Most NoSQL databases are driven by the need to run on clusters, and this is certainly true of those that were talked about during the initial meetings about NoSQL. This has an effect on their data model as well as their approach to consistency. Relational databases use ACID transactions to handle consistency across the whole database. This inherently clashes with a cluster environment, so NoSQL databases offer a range of options for consistency and distribution.

NoSQL databases operate without a schema, allowing freely adding fields to database records without having to define any changes in structure. This is particularly useful when dealing with non-uniform data and custom fields which force relational databases to use names like customField6 or custom field tables that are awkward to process and understand.

March 2009 InfoWorld article entitled *NoSQL Databases Break All the Old Rules* (Wayner, 2009) described a set of data storage and access technologies that had been developed by Web giants like Google and Amazon and were being adversely impacted by the open-source software community. Since then, the term *NoSQL database* has become something of a battle-cry, a marketing meme, and a label attached to more than a hundred products, some of which pre-date the label by many years (Edlich, 2012). Anything that defines itself by what it's not is literally unbounded, so it's unsurprising that even the meaning of the term—No SQL or Not only SQL—is also morphing.

2.3.2 Relational Database Management Systems (RDBMS). A relational database management system (RDBMS) is a software application tool. It provides the capabilities of controlling, managing, and administrating an organization's data storage. It is the tool that utilized to create a relational database using Structured Query Language (SQL). Atomicity, Consistency, Isolation, Durability (ACID) is a concept that ensures the reliability of database transactions processing. A transaction is a database operation (Brewer, 2000).

- **A**tomicity: Everything in a transaction succeeds lest it is rolled back. SQL carries this principle internally. An INSERT statement inserts an entire set of rows into a table; a DELETE statement deletes an entire set of rows from a table; an UPDATE statement deletes and inserts entire sets (Celko, 2013).
- **C**onsistency: A transaction cannot leave the database in an inconsistent state. In SQL this means that all constraints are TRUE at the end of a transaction. This can be because the new state of the system is valid, or because the system was rolled back to its initial consistent state.
- **I**solation: One transaction cannot interfere with another.
- **D**urability: A completed transaction persists, even after applications restart.

Beyond ACID, there are three core systemic requirements that exist in a special relationship when it comes to designing and deploying applications in a distributed environment (Brewer, 2000). The three requirements are: Consistency, Availability and Partition Tolerance or CAP.

- **C**onsistency: A service that is consistent operates fully or not at all.
- **A**vailability: means just that the service is available to operate fully or not at all.

- **P**artition Tolerance: means that a given system continues to operate even with data loss or system failure.

2.3.3 Key Advantages of NoSQL. The key advantages of NoSQL include:

- Schema-less: Almost all NoSQL implementations offer schema-less data representation.
- Less time
- Speed
- Scalability

2.3.4 RDBMS VS. NoSQL. Many aspects compared between the traditional RDBMS solutions and NoSQL databases. Table 1 illustrates the main difference between the traditional relational databases management systems (RDBMS) and NoSQL in terms of flexibility, query complexity, and scalability. These features are qualitative in nature the draws the outline of each type key characteristics.

Table 1.

Comparison between RDBMS and NoSQL

RDBMS	NoSQL
Identify actors: The first step in the traditional approach is to identify various actors in the application. The actors can be internal or external to the application.	**Schema flexibility:** store data as columns as opposed to rows in RDBMS. This allows flexibility of adding one or more columns as required, on the fly. Similarly, document stores that allow storing semi-structured data are also good options.

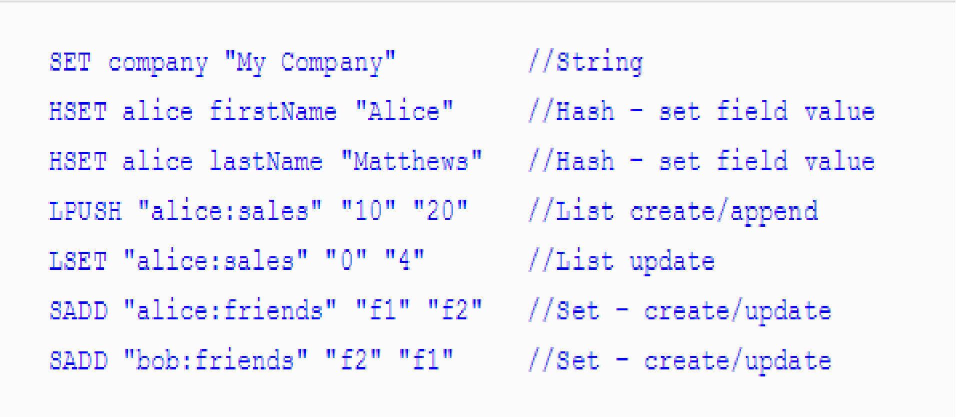
Table 1.

Cont.

Define models: Once the actors are identified, the next step is to create models. Typically, there is many-to-one mapping between actors and models, that is, one model may represent multiple actors.	**Complex queries**: NoSQL databases do not have support for relationships or foreign keys. There are no complex queries. There are no JOIN statements.
Define entities: Once the models and the object-relationships—by way of inheritance and encapsulation—are defined, the next step is to define the database entities	**Data update**: Data update and synchronization across physical instances are difficult engineering problems to solve.
Define relationships: One of more important steps is to be able to well define the relationship between the entities.	**Scalability:** NoSQL solutions provide greater scalability for obvious reasons. A lot of complexity that is required for transaction oriented RDBMS does not exist in ACID non-compliant NoSQL databases.

2.3.5 NOSQL Categories. In this section, the main NoSQL databases and their application are identified. Worth noting that Key-values databases and Document databases are more mature than the Graph databases in terms of years of development, commercial brands, the variety of vendors, and years of accumulated knowledge.

2.3.5.1 Key-Value Stores. Key-value stores are the simplest NoSQL data stores to use from an API perspective. The client can either get the value for the key, put a value for a key, or delete a key from the data store. It allows the storage of a value against a key. Similar to a document store, there is no need for a schema to be enforced on the value. However, there are few constraints that are enforced by a key-value store (Vaish, 2013). Figure 5 shows example of a set value against a key.



```
SET company "My Company"            //String
HSET alice firstName "Alice"        //Hash - set field value
HSET alice lastName "Matthews"      //Hash - set field value
LPUSH "alice:sales" "10" "20"       //List create/append
LSET "alice:sales" "0" "4"          //List update
SADD "alice:friends" "f1" "f2"      //Set - create/update
SADD "bob:friends" "f2" "f1"        //Set - create/update
```

Figure 5. Example of Set a value against a key

Advantages of Key-value Stores:

- Consistency
- Transactions
- Optimized Querying

Key-value stores Databases are good for:

- Session information
- User profiles
- Shopping Cart

2.3.5.2 Document Databases. The database stores and retrieves documents, which can be XML, JSON, BSON, etc. These documents are self-describing, hierarchical tree data structures which can consist of maps, collections, and scalar values. The documents stored are similar to each other but do not have to be exactly the same. Document databases store documents in the value part of the key-value store; think about document databases as key-value stores where the value is examinable (Sadalage & Fowler, 2012). Even though the documents do not follow a strict schema, indexes can be created and queried.

Advantages of Document Databases:

- Consistency
- Easy use for web-based applications
- Schema-less
- Easy to define
- Scaling

Document Databases are good for:

- Event logging
- Content Management System (CMS)
- Web analytics
- E-commerce platforms

2.3.5.3 Column Oriented Databases. Column stores have been in development since early DBMS days. TAXIR, a biology information-retrieval-focused application, was the first application using column-oriented stores way back in 1969 (Vaish, 2013).

2.3.6 Graph Databases. A new kind of database called a “graph’ database has recently emerged. A NoSQL database is a graph databases are noticeably post-relational data stores, because they evolve several database concepts much further while keeping other attributes. They provide the means of storing semi-structured data but extremely connected data efficiently and allow the user to query and traverse the connected data at a very high speed and efficiency.

Graph databases, as Figure 6 illustrates usually consist of nodes connected with directed and labeled relationships. In property graphs, both nodes and relationships can hold arbitrary key/value pairs. Graphs form a sophisticated network of those elements and ease the process to model domain and real-world data. The graph is a structure close to the original structure which is a big difference from traditional databases. Unlike relational databases, which rely on well-defined schemas to model data, graph databases are schema-free and put no constraints onto the data structure which open more space for database architects to be creative. Relationships can be added and changed easily, because they are not part of a schema but rather part of the actual data. Figure 6 illustrates an example of how the nodes and relationships connect.

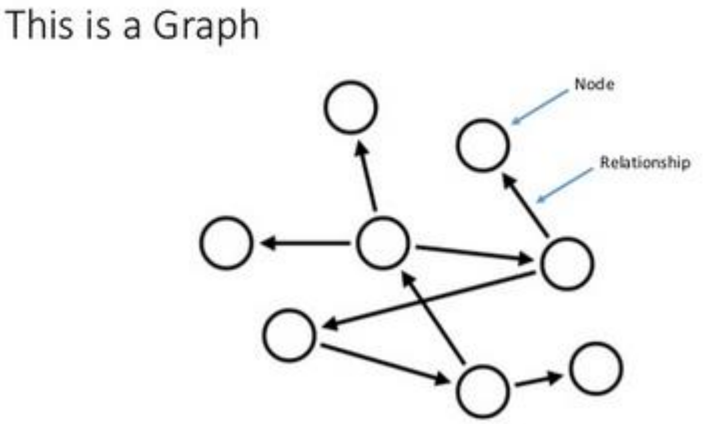


Figure 6. Visual Example of an actual Graph data model

Figure 7 shows that the graph involves equation vertices and edges. Vertices (Nodes) includes the following: unique identifiers, outgoing edges, incoming edges, and key/value pairs. While the edges (relationships) consist of: Unique identifiers, outgoing vertex (node), incoming vertex (node), and again key/value pairs.

G = (V, E)
Graph
Vertex
Edge

Figure 7. Graph theory equation

Graph databases are based on algebraic equations while relational databases are based on sets, graph databases are based on graph theory, a branch of discrete mathematics. A graph database management system (henceforth, a graph database) is an online database management system typically contains the commands that Create, Read, Update, and Delete. Graph databases are generally built for use with transactional (OLTP) systems. Accordingly, they are normally optimized for transactional performance, and engineered with transactional integrity and operational availability in mind.

A graph has two things in it. There are edges (or arcs) and nodes (or vertices); the edges are drawn as lines that connect nodes, which are drawn as dots or circles. Nodes are abstractions. They usually (not always) model what would be an entity in a RDBMS. In fact, some of the power of graph theory is that a node can model a sub graph. A node may or may not have “something inside it” (electrical components) in the model; it can just sit there (bus stop) or simply be (transition state). A node is not an object. Objects have methods and local data inside them. Other examples of graphs are:

- Schematic maps: the nodes are the bus stops, towns, and so forth.
- Circuit diagrams: the nodes are electrical components.
- State transitions: the nodes are the states.

Edges or arcs connect nodes. They are drawn as lines that may or may not have an arrow head on them to show a direction of flow when the edges have arrows, the graph is called a "directed" graph otherwise it is called un-directed graph. They may also contain weights denoting the strengths of the relationship between the nodes. In schematic maps, the edges are the roads. They can have a distance or time on them. In the circuit diagrams, the edges are the wires that have resistance, voltage, etc. Likewise, the abstract state transitions are connected by edges that model the permitted switch paths.

In one way, edges are more fun than nodes in graph theory. In RDBMS models of data, an unspecified single relationship between tables exists in the form of a REFERENCES clause. In a graph database, there are multiple edges of different kinds between nodes.

At the highest level of abstraction an edge can be directed or undirected. In terms of maps, these are one-way streets; for state transitions, this is prior state–current state pairs and so forth. Undirected graphs is preferable since the math is easier and there is often an inverse relationship of some sort (e.g., "employer–manager" in the case of Jose Mourinho and Jorge Valdano).

2.3.6.1 Graph Databases Pros and Cons. There are several pros and cons associated with graph databases, the pros are a powerful data model that is not limited with any schema constrains, connected data that have many layers and levels of details and properties, and locally indexed inside the database architect, and easy and flexible to query. Nevertheless graph databases do have multiple cons, too much sharding either horizontal or vertical partition of the data inside the graph database, and a very diverse approach from relational database concept

2.3.6.2 Graph Database Domains. Many applicable industries or solution landscape can be deployed using the graph databases, impact analysis (network, software), logistics, supply chain for package routing, recommendation systems (dating, social…etc.), master data management, financial especially fraud detection.

In this section we have highlighted the main challenges in designing graph databases. To this end, three specific challenges were targeted by this research study – data models, storage models and index structures (Pardede & Sakr, 2011). Graph databases has attracted vast research attention, but has still eluded strong foundations or an overarching best practice or design principles. Research and commercial interest in graph databases is still increasing as of this writing. As the amount of data grows and the need for identifying patterns and extracting semantics becomes stronger, graph data management becomes an imminent challenge (Pardede & Sakr, 2011).

2.4 Neo4j

Neo4j is the leading application of a property graph database. It is written mainly in Java and leverages a custom storage format and the facilities of the Java Transaction Architecture (JTA) to provide XA transactions. The Java API offers an object-oriented approach to manipulate the nodes and relationships of the graph (Pollack; Risberg; Hunger; Brisbin; & Gierke, 2012). Traversals are expressed with a fluent API. Being a graph database, Neo4j offers a number of graph algorithms like: Shortest path, Dijkstra, and A* out of the box.

Neo4j integrates a transactional, pluggable indexing subsystem that uses Lucene as the default. The index is used primarily to locate starting points for traversals. Its second use is to support unique entity creation (Pollack; Risberg; Hunger; Brisbin; & Gierke, 2012). Neo4J provides a web interface for statistics about the database. In the data browser, it is possible to search for nodes by ID, or with index lookups, and with cypher queries, and switching to the graph visualizer with the right-hand button for visually exploring the graph. The console allows the user to enter Cypher statements directly or issue HTTP requests. The other great advantage is that Neo4j an open source product, hence it has a comprehensive and active ecosystem. Neo Technology, is the company sponsoring the development of Neo4j. Neo4j also comes with a variety of programming language drivers available. There are libraries for many programming languages for both embedded and server deployment mode.

2.4.1 List of Programming Language Drivers. Neo4j have many addable programming language driver that includes: PHP Cypher, Python, R Neo4J, Perl, Ruby, Node.JS, Scala, and JavaScript.

Neo4J is considered a property graph, which is a “directed, labeled, attributed multi-graph” that exposes three building blocks: nodes, typed relationships, and key-value properties on both nodes and relationships. Another graph model is RDF Triples, which is a URI-centered subject-predicate-object triples as pioneered by the semantic web movement. Finally there’s HyperGraph, which is a generalized graph in which a relationship can connect an arbitrary number of nodes (compared to the more common binary graph models) (McKnight, 2013).

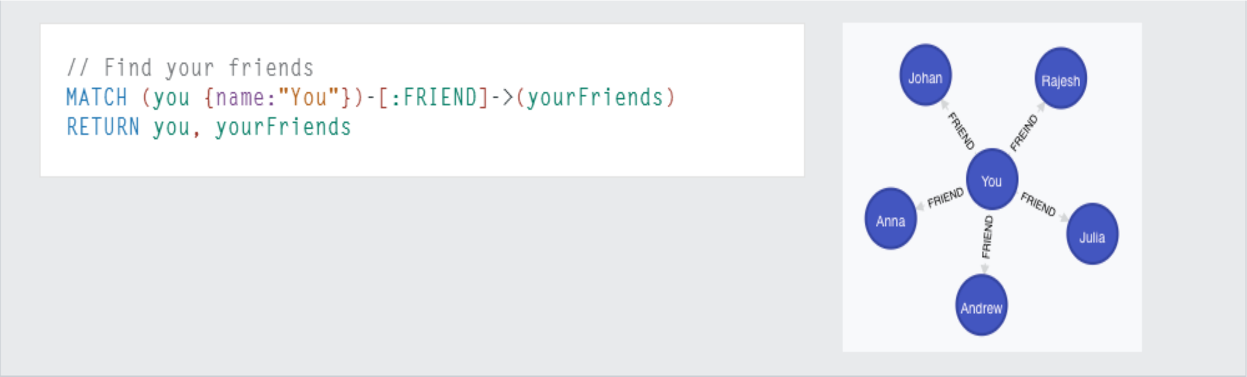
Hypergraph is a generalized graph model in which a relationship (called a hyper-edge) can connect any number of nodes. More specifically, while the property graph model permits a relationship to have a single start and end node, the hypergraph model allows any number of nodes at a relationship’s start and end (McKnight, 2013).

Neo4J has a declarative Cypher query language, which makes it easier to get started for everyone who knows SQL from working with relational databases. Developers as well as operations and business users can run ad-hoc queries on the graph for a variety of use cases. Cypher draws its inspiration from a variety of sources:

- SQL

- SparQL
- ASCII-Art
- Different functional programming

Cypher was built to create matching patterns in a graph that are bound to specified nodes and relationships and allows additional filtering and indexing of the results. Cypher has data manipulation features that allow us to modify the graph. Cypher query parts can be chained (piped) to enable more advanced and powerful graph operations. Figure 8 depicts a cypher query code and the results after executing this query



```
// Find your friends
MATCH (you {name:"You"})-[:FRIEND]->(yourFriends)
RETURN you, yourFriends
```

Figure 8. Example of Cypher code

The core concept is that the user describes the patterns to be matched in the graph and find starting points. The Neo4J database engine then efficiently matches the given patterns across the graph, enabling users to define sophisticated queries like “find me all the people who have friends or co-workers who have recently bought gadget products.”, Cypher supports filtering, grouping, and paging. Cypher allows easy creation, deletion, update, and graph construction.

- Nodes: Cypher uses ASCII-art to represent patterns. Nodes are surrounded with parentheses which look like circles, e.g. `(node)`.

- Relationships: The problem with the Cypher snippets is that they don’t contain information about the relationship between the nodes. So the need for describing the types of relationships in Cypher queries is essential (Onofrio Panzarino, 2014).
- Labels and Cypher Query Language: Labels give a very convenient feature by allowing to group our nodes. The Cypher query language is comprised of several distinct clauses.

Querying a graph involves these commands:

- **MATCH**: Matches the given graph pattern in the graph data.
- **WHERE**: Filters using predicates or anchors pattern elements.
- **RETURN**: Returns and projects result data, also handles aggregation.
- **ORDER BY**: Sorts the query result.
- **SKIP/LIMIT**: Paginates the query result.

Updating the graph involves these commands:

- **CREATE**: Creates nodes and relationships.
- **MERGE**: Creates nodes uniquely.
- **CREATE UNIQUE**: Creates relationships uniquely.
- **DELETE**: Removes nodes, relationships.
- **SET**: Updates properties and labels.
- **REMOVE**: Removes properties and labels.
- **FOREACH**: Performs updating actions once per element in a list.

- **WITH**: Divides a query into multiple, distinct parts and passes results from one to the next.

Cypher Principles and Capabilities

- Cypher matches patterns of nodes and relationship in the graph, to extract information or modify the data.
- Cypher has the concept of identifiers which denote named, bound elements and parameters.
- Cypher can create, update, and remove nodes, relationships, labels, and properties.
- Cypher manages indexes and constraints.

Neo4j Advantages

- Close to the object model
- Pre materialize relationships
- Traversals in linear time
- Sparse, heterogeneous data and schema free
- Local queries
- User Interface is super friendly

2.5 Neo4J Applications

The graph data model in general and Neo4J specifically renowned for the power and flexibility, and its essential expressiveness. The graph databases application in a real-world problem can be seen as the fast performance of the queries and responsiveness helps organizations with their online transactional systems. Graph databases also ideal for quick development cycles, finally Neo4J is “Enterprise” ready in terms of capabilities, performance, and characteristics.

2.5.1 Social Science Application. Social applications allow organizations to gain competitive and operational advantage by leveraging information about the connections between people, together with discrete information about individuals, to facilitate collaboration and flow of information, and predict behavior.

As Facebook’s use of the term social graph implies, graph data models and graph databases are a natural fit for this overtly relationship-centered domain (Robinson; Webber; & Eifrem, 2013).

By understanding who interacts with whom, how people are connected, and what representatives within a group are likely to do or choose based on the aggregate behavior of the group, Neo4J can generate tremendous insight into the unseen forces that influence individual behaviors.

2.5.2 Master Data Management. Master data is data that is critical to the operation of a business, but which itself is non-transactional. Master data includes data concerning users, customers, products, suppliers, departments, geographies, sites, cost centers, and business units.

Graph databases don’t provide a full MDM solution; they are, however, ideally applied to the modeling, storing, and querying of hierarchies, master data metadata, and master data models. Such models include type definitions, constraints, relationships between entities, and the mappings between the model and the underlying source systems (Robinson; Webber; & Eifrem, 2013).

2.5.3 Logistics and Supply Chain. Logistics related applications of graph databases range from calculating routes between locations in an abstract network such as a road or rail network, airspace network, or logistical network to spatial operations such as find all points of interest in a bounded area, find the center of a region, and calculate the intersection between two or more regions (Robinson; Webber; & Eifrem, 2013).

Logistics operations depend upon specific data structures, ranging from simple weighted and directed relationships, through to spatial indexes that layout multidimensional attributes using tree map data structures.

2.5.4 Recommended Systems. Effective recommendations are a prime example of generating end-user value through the application of an inferential or suggestive capability. Recommendation algorithms establish relationships between people and things: other people, products, services, media content—whatever is relevant to the domain in which the recommendation is employed (Robinson; Webber; & Eifrem, 2013). As nodes and relationships are created in the system, they can be used to make recommendations like "when invoicing this item, these other items are usually invoiced." Or, it can be used to make recommendations to travelers mentioning that when other visitors come to Barcelona they usually visit Antonio Gaudi's creations (Sadalage & Fowler, 2012).

2.5.5 Fraud Detection Systems. The principle is quite simple: in many cases, the fraud of a particular nature is not defined by one transaction only, but by a chain of transactions that have their specific characteristics and that need to be compared to one another to see if they really do constitute a case of fraud. That are based on linked intelligence. Unlike the traditional relational database technologies which requires modeling the graph above as a set of tables and columns, and then carrying out a series of complex joins and self joins. This type of queries are very complex to build and expensive to run on a traditional RDBMS. Scaling them in a way that supports real-time access poses significant technical challenges, with performance becoming exponentially worse not only as the size of the ring increases, for that reason Graph databases have emerged as an ideal tool.

CHAPTER 3

RESEARCH DESIGN AND PROCEDURES

Based on the research question and purpose, the study of this research follows an exploratory approach in which a mixed method is established for examining and evaluating the prototype. A mixed method encompasses the quantitative and qualitative approaches as subjective and objective procedures (Mahmud, 2006). In a mixed method, the study of this research tends to build the knowledge claim on practical arguments such as outcome-directed, or problem-centered (Creswell, 2003). In a quantitative approach, the knowledge is developed via the implementation of postpostivist claims. Examples include shrinking to the questions of this research, applying measurement and remark and testing the theory. Postpostivist viewpoint is the thinking after positivism, disputing the typical certainty of the knowledge of our research (Creswell, 2003).

Since the study solution of this research provides an interactive system to accelerate information retrieval, mixed method can be absolutely relevant to such system. It is forceful and practical everywhere in the computing area where numerous jobs may be engaged in an application. The advantages of mixed method include presenting a huge data, and practical environment. Also, it provides portability (Mahmud, 2006).

In this thesis, a proof of concept, also known as a vertical prototype, implements a slice of application functionality from the user interface through all the technical services layers. It works as a real system at all levels of the system implementation. It is developed to validate the concept of the proposed architectural approach and to evaluate the proposed database schema.

To make the results meaningful, such prototype is constructed by using production tools in a production-like operating environment. Agile development projects sometimes refer to a proof-of-concept prototype as a “spike.” (Alan, 1993).

A throwaway prototype is built to answer questions, resolve uncertainties, and improve requirements quality (Alan, 1993). It emphasizes quick implementation and modification over robustness, reliability, performance, and long-term maintainability. For this reason, low-quality code is not allowed from a throwaway prototype to migrate into a production system (Beatty, Wiegers, 2013).

In contrast to a throwaway prototype, an evolutionary prototype provides a solid architectural foundation for building the product incrementally as the requirements become clear over time (McConnell, 1996).

Agile development provides an example of evolutionary prototyping. Agile teams construct the product through a series of iterations, using feedback on the early iterations to adjust the direction of future development cycles. This is the essence of evolutionary prototyping.

In contrast to the quick-and-dirty nature of throwaway prototyping, an evolutionary prototype must be built with robust, production-quality code from the outset. Therefore, an evolutionary prototype takes longer to create than a throwaway prototype that simulates the same system capabilities. An evolutionary prototype must be designed for easy growth and frequent enhancement, so developers must emphasize software architecture and solid design principles. There’s no room for shortcuts in the quality of an evolutionary prototype (McConnell, 1996). Table 2 represents the difference between the two prototype approaches.

Table 2.

Difference between approaches

	Throwaway	**Evolutionary**
Mock-up	• identify user/functional requirements	• Implement main users requirements
	• locate the missing functions	• Apply the extracted requirement into the system/architect
	• Explore user interface approaches	• Accommodate the system to continuous changing business requirements
Proof of Concept	• Demonstrate technical feasibility	• Implement and grow core multi-tier functionality and communication layers.
	• Evaluate performance	• Implement and optimize core algorithms
	• Acquire knowledge to improve estimates for construction.	• Test and tune performance.

Figure 9 explains several possible ways to combine the various prototypes. For example, the knowledge gained from a series of throwaway prototypes can be used to refine the requirements, which might then implement incrementally through an evolutionary prototyping sequence (Beatty; Wiegers, 2013).

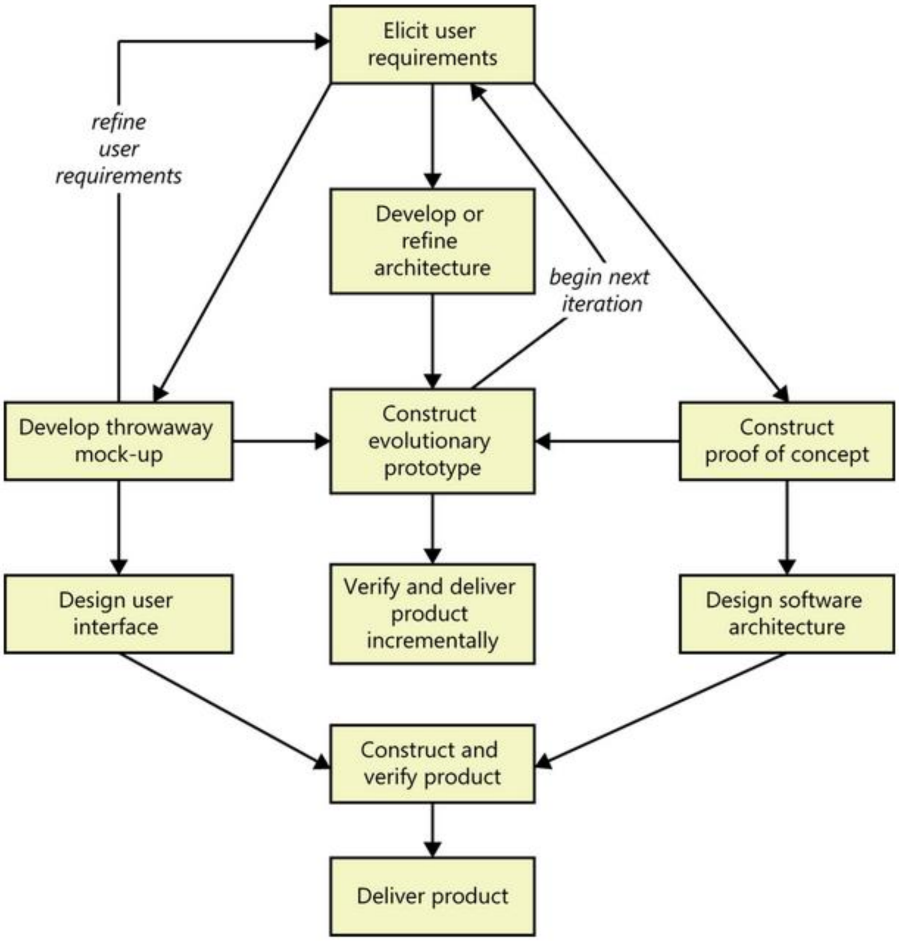

Elicit user requirements
refine user requirements
Develop or refine architecture
begin next iteration
Develop throwaway mock-up
Construct evolutionary prototype
Construct proof of concept
Design user interface
Verify and deliver product incrementally
Design software architecture
Construct and verify product
Deliver product

Figure 9. Incorporation of prototyping into the software development process

3.1 The Prototype Model

In the study of this research, the prototyping model is an evolutionary procedure that replaces some of the Software Development Life Cycle’s phases. The purpose of this process is based on the developer, users, and customers’ communication channel (Galin, 2004). The user receives the prototypes to examine it and provides feedbacks. The developer keeps on developing a more advanced prototype. Again, the user receives it to examine the new developments and provides further feedbacks. Such procedure is repeated either until the study solution satisfies the specified requirements or the prototyping purpose is accomplished.

Thus, the essential features of the prototyping process can be identified by a set of characterization. This set of characterization consists essentially of several iterative cycles. The initial prototype is an executable software model that is constructed based on either an initial selection of functions or based on user's needs that have been identified. A demonstration or actual usage of the prototype allows review by users. User's suggestions, criticisms and enhancements result in revision of the prototype. This cycle is continued for each revised version. The prototyping process is completed when the desired objective is attained (El-Bathy, 2011).

Evolutionary prototyping is a procedure that makes a prototype the eventual operational system such that recreation or redevelopment of some of the system's pieces may be needed by applying other methodologies (Anton, A. & Carter, R. & Earp, J. & Williams, L., 2001).

Neumann and Jenkins (1982) characterize a prototyping as a four step, iterative procedure involving users and developers:

1. User's basic needs are identified;
2. A working prototype is developed
3. The working prototype is then implemented and used
4. The prototyping system is revised and enhanced

This process undergoes several iterations and both steps three and four are repeated until the user accepts the system.

Maner (1997) states that prototyping is a model that represents a designed application which can be changed or enhanced. The application's functions and interfaces are contained within the prototyping. The processes of Maner's prototyping are subject to many variations. Maner (1997) shows these processes in Figure 10 as follows:

1. Analyzing the customers or the users' requirements
2. Clarifying the initial requirements by building a low fidelity prototype
3. Conducting iteration processes in order to re-identify, re-plan, re-examine the level of the users' satisfaction based on their needs.
4. Freeze the specifications
5. Completing the system's development

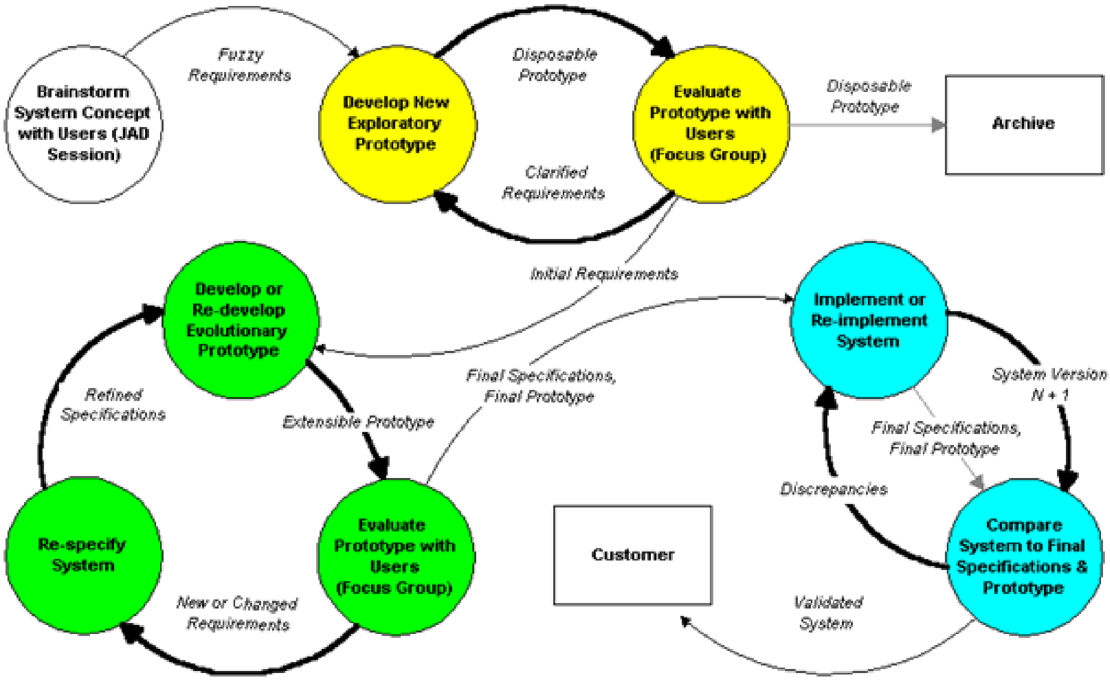

Brainstorm System Concept with Users (JAD Session)
Fuzzy Requirements
Develop New Exploratory Prototype
Disposable Prototype
Evaluate Prototype with Users (Focus Group)
Clarified Requirements
Disposable Prototype
Archive
Initial Requirements
Develop or Re-develop Evolutionary Prototype
Refined Specifications
Extensible Prototype
Re-specify System
Evaluate Prototype with Users (Focus Group)
New or Changed Requirements
Final Specifications, Final Prototype
Implement or Re-implement System
System Version N + 1
Final Specifications, Final Prototype
Discrepancies
Compare System to Final Specifications & Prototype
Customer
Validated System

Figure 10. Walter Maner's Iterative Prototype Process

3.2 The Prototype High Level Use Case

Figure 11 reflects the general picture of Graphical Database architect (GDA) for Clinical Trial high level use case. The diagram defines the main activities of the system. The component of the diagram explains the interaction between the user/management and the system.

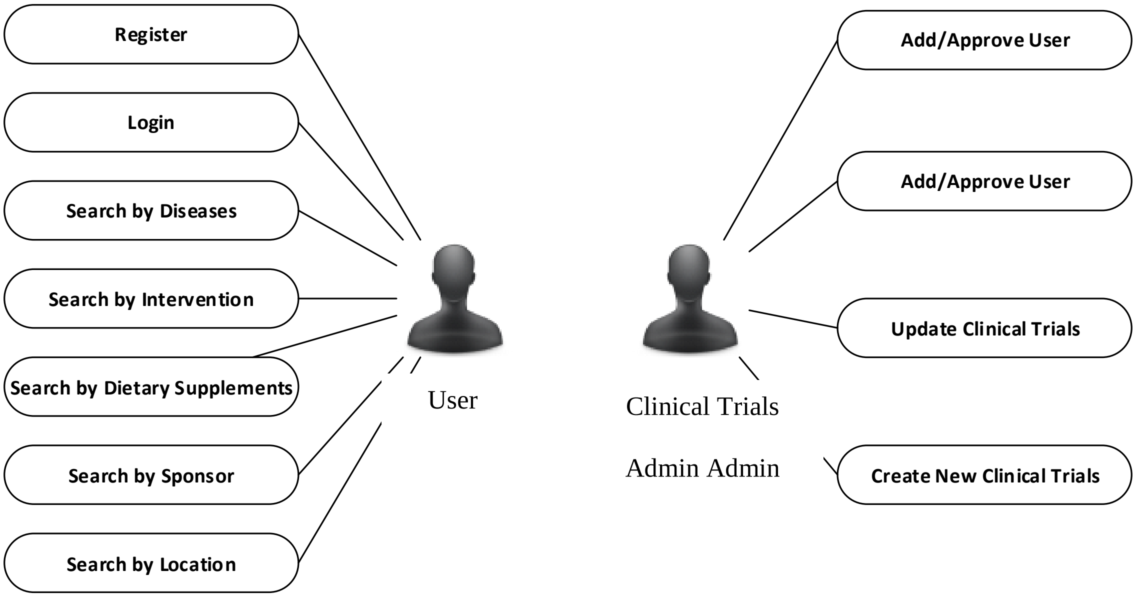


Figure 11. High Level User Cases

3.3 Prototype Requirements

User Interface (UI) is built as a web interface on Windows 8 platform. This UI forms the starting point of the Graphical Database architect (GDA) for Clinical Trial, and further requirements will be built onto it. The User Interface is then added to the prototype requirements Specification document.

The Graphical Database architect (GDA) for Clinical Trial is intended to permit the development of a set of means that address performance, capabilities, and functionality concerns.

The prototype User Interface (UI) is identified as abstract of front end interfaces that represent the backend of the architect of the model and its implementation. The architecture operates as a model that can be used for showing the establishment of several instances of the system. It enables implementing the services by several application components. In this study, the Evolutionary Rapid modeling is a process where model the data aspects of a system iteratively and incrementally, just like all other aspects of a system, to ensure that the database schema evolves in step with the application code.

Table 3.

The GDA prototype functions

Functionality	**Type**
Register	Essential
Login	Essential
Search for trial by diseases	Essential
Search for trial by intervention	Essential
Search for trial by dietary supplement	Essential
Search for trial by sponsor	Essential
Search for trial by location	Essential
Create new clinical trial	Essential
Updated an existing clinical trial	Essential

3.4 GDA for Clinical Trials Architecture Prototype

This section explains the prototype architecture of Graphic Database Architecture (GDA) for Clinical Trials. Figure 12 describes the three-tiered web-enabled prototype architecture.

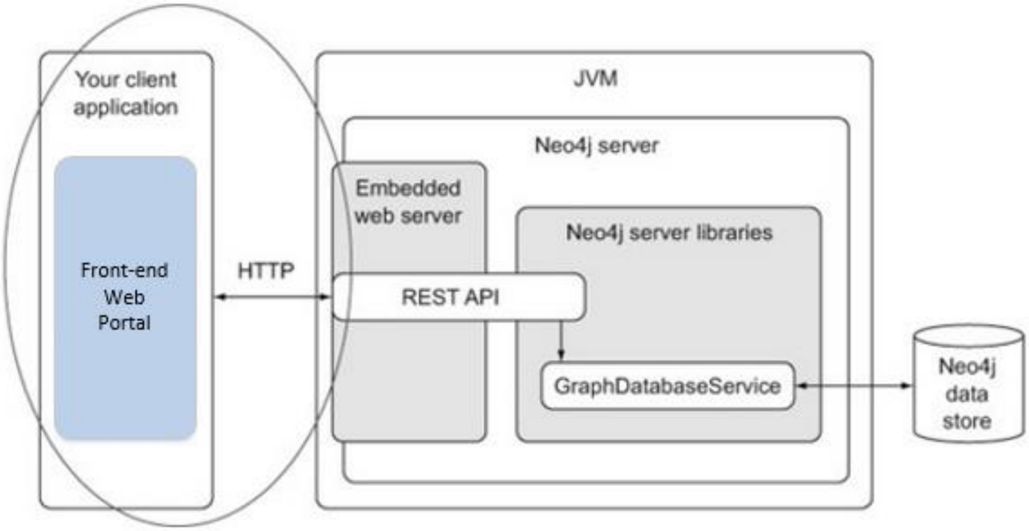


Figure 12. Neo4j Architecture for GDA prototype

Portrays the core components involved in a running Neo4j in server mode, along with a client accessing the server using the standard, and REST API. The Neo4j server is itself simply a JVM-based application. Under the covers, it provides its functionality by wrapping an appropriate instance of the “GraphDatabaseService” interface (EmbeddedGraphDatabase or HighlyAvailableGraphDatabase), exposing the functionality to the outside world through a well-defined REST interface. To be able to listen to and react to REST requests made to the server by clients, all Neo4j server instances will start up an embedded web server.

Neo4j can be run in two different modes:

- An embedded database in a Java application
- A standalone server via REST

In any case, this choice does not affect querying and working with the database. It's an architectural choice driven by the nature of the application (whether a standalone server or a client server), performance, monitoring, and safety of data (Panzarino, 2014).

3.5 Tools for the Prototype Implementation:

This section describes software interfaces and development tools that are used for the prototype. These tools are compatible with windows environment, in both terms of designing and coding. In terms of design, the Microsoft Visio is used to design the Prototyping Model and Prototyping Architecture.

3.5.1 Software Interfaces. The following is a list of software used:

- Operating System – Microsoft Windows 8
- The backend server is a Java web application, running an embedded Jetty servlet container.
- Database Server – Neo4j

3.5.2 Development Tools. The Tools that are used for prototype implementation

- Graph Gist
- Graph ETL
- Neo4j Community edition

3.6 The Prototype Evaluation Criteria

The remainder of this subsection describes the walkthrough methodology. Walkthrough evaluates the GDA for Clinical Trials architecture of extensive, complicated, and heterogeneous system. Walkthrough supports dynamic development such that the architectural demonstration can be recommended precisely during walkthroughs with the project’s stakeholders (Haynes & Skattebo & Singel & Cohen & Himelright, 2006).

The models based on cognitive architectures have been used in human-computer interaction in three ways: to predict task performance times and errors, to assist users in task performance, or to substitute users in testing (John, 1998; Ritter et al., 2000)
An integrative approach to UI design is an approach based on the principles that unify cognition and other elements of interactive behavior. These principles also unify theories on interactive behavior and their methodology with UI design (Kljajevic, 2008)

An informal walkthrough is mainly a helpful method for separating and determining ambiguity that is related to unclear system specifications. A formal walkthrough can be modified to develop a recursive evaluation process as the informal walkthrough advances and produces detailed outputs (Haynes & Skattebo & Singel & Cohen & Himelright, 2006).

The techniques of walkthrough are approved as experimental assessment approaches to evaluate system application usability (Rowley & Rhoades, 1992). A cognitive structured walkthrough method theoretically examines the procedure based on a number of questions. These questions help the interface designer concentrate on the important characteristics for solving the theory's problem (Lewis & Poison & Wharton & Rieman, 1990).

A groupware walkthrough presents an independent adaption of cognitive walkthrough method. It is a technique that is adjusted for groupware usability inspection. This technique is also applicable for real work situation. It is usable in developing software system in its early phases. It reveals the system usability problems (Pinelle, & Gutwin, 2002).

For a single user of a software system, cognitive walkthrough is considered a general evaluation methodology since it enables the software evaluation during its early development's phases. Cognitive walkthrough also doesn't need a prototype that is functioning. It describes the system's users along with their knowledge. It also depicts the system user's roles. Via a walkthrough, the evaluators walk through these roles to ensure that the user performs the exact role based on the prototype (Pinelle, & Gutwin, 2002).

The walkthrough methodology of cognitive and groupware keep information of the system’s users and their tasks externally. The designer of the system interface is not responsible for the other details about the work situation. The evaluators of the cognitive and groupware walkthrough methodologies are able to briefly deliver information about the tasks without any details (Pinelle, & Gutwin, 2002).

The walkthrough technique that is developed by groupware can be performed the same as the techniques of discount usability at a moderate and low cost. Multi-user systems can be evaluated as well through groupware walkthrough technique in conjunction with groupware heuristic evaluation at low cost. In the initial development process, groupware walkthrough method is used to enable the system’s developers to apply formative evaluation during the design phase. During the system evaluation phase, the walkthrough technique guarantees the discovery of the majority of the problems related to the automatic usability such that a located examination can concentrate on the problems of the highest level. The examiners will be able to evaluate the interfaces that are related to real roles and real customers (Pinelle, & Gutwin, 2002).

The study of this research plans to carry out a walkthrough technique that examines the prototype. The walkthrough method provides a comprehensive evaluation the phases of the research. These phases of this thesis includes the planning problem identification, the research planning, the literature review, the research proposed solution, and the research testing.

The walkthrough participants of this study would be clinical trials investigators, medical professionals, pharmaceutical professionals and students. The aim of the walkthrough is evaluate the prototype to ensure that it satisfies requirements of this research. They will verify that the solution is visualized correctly. They also will have the chance to verify that the documented findings solve the research problem.

3.7 The Research Analysis

The analysis of this research is planned to review recent knowledge about Graphical Database Architecture for Clinical Trials, discover knowledge gaps, and identify future research target. The research analysis results are projected to help solve the problems of clinical trials by accelerating information retrieval processes in Medical and Healthcare organization.

The analysis reviews recent literature about the Medical and Healthcare industries' information retrieval lifecycle. In general, the analysis goal is identifying the required information for accelerating information retrieval processes based on the alignment between IT and the business strategy. The goals of the analysis provide and define:

1. The main knowledge gaps in the business
2. The key strategies to accomplish the goals
3. The research requirements and limitations
4. The publishing and newspaper users of the research solution
5. The knowledge gaps affect the research analysis.

The conceptual models are the core of the prototype development iteration and these models should be integrated in the analysis stage of a lifecycle model (El-Bathy & Gloster, 2012). The study of this research has focused on the following:

3.7.1 The Method of Data Collection. In data collection step, data from Clinical Trials website is collected. The collected data is stored into database for additional process. The collected data from Clinical Trials site is semi-structured data uses Neo4j. Clinical Trials data is a semi-structured data as Figure 13 illustrates data that doesn't reside in a relational database but that does have some organizational structure properties that make it easier to analyze.

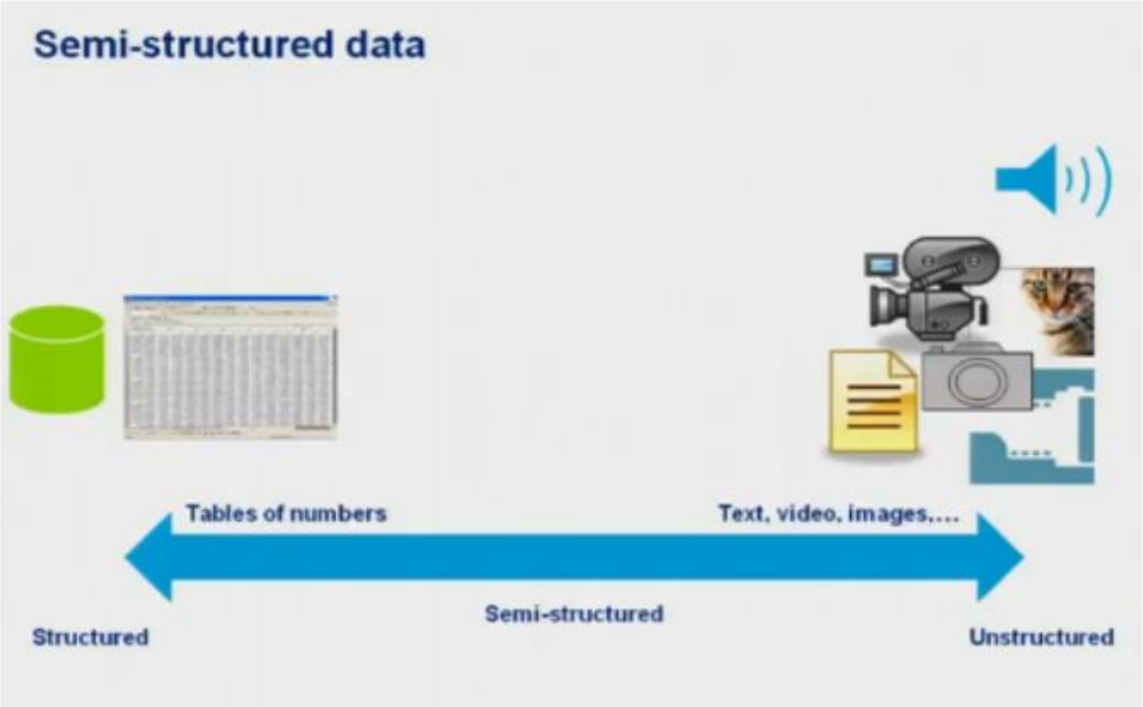


Figure 13. Data Types Landscape

In this research using Clinical Trials data is used to build the ontology of the graphical database. The ClinicalTrials.gov results database was launched in September 2008 to implement Section 801 of the Food and Drug Administration Amendments Act of 2007 (FDAAA 801), which requires the submission of "basic results" for certain clinical trials, generally not later than 1 year after their Completion Date.

The submission of adverse event information was optional when the results database was released but became required in September 2009. Results information for registered and completed studies is submitted by the study sponsor or principal investigator in a standard, tabular format without discussions or conclusions. The information is considered summary information and does not include patient-level data. The results information that is submitted includes the following:

- **Participant Flow.** A tabular summary of the progress of participants through each stage of a study, by study arm or comparison group. It includes the numbers of participants who started, completed, and dropped out of each period of the study based on the sequence in which interventions were assigned.

- **Baseline Characteristics.** A tabular summary of the data collected at the beginning of a study for all participants, by study arm or comparison group. These data include demographics, such as age and gender, and study-specific measures (for example, systolic blood pressure, prior antidepressant treatment).

- **Outcome Measures and Statistical Analyses.** A tabular summary of outcome measure values, by study arm or comparison group. It includes tables for each prespecified Primary Outcome and Secondary Outcome and may also include other prespecified outcomes, post hoc outcomes, and any appropriate statistical analyses.

- **Adverse Events.** A tabular summary of all anticipated and unanticipated serious adverse events and a tabular summary of anticipated and unanticipated other adverse events exceeding a specific frequency threshold. For each serious or other adverse event, it includes the adverse event term, affected organ system, number of participants at risk, and number of participants affected, by study arm or comparison group. Then ClinicalTrials.gov organizes information for each registered study as an integrated unit, displaying the study protocol information and, if available, the corresponding results information on the same page under different tabs. (Clinical Trials Website)

3.7.2 XML Data. The eXtensible Markup Language (XML) is increasingly finding acceptance as a standard for storing and exchanging structured and semi-structured information over internet (Conrad, 2000)

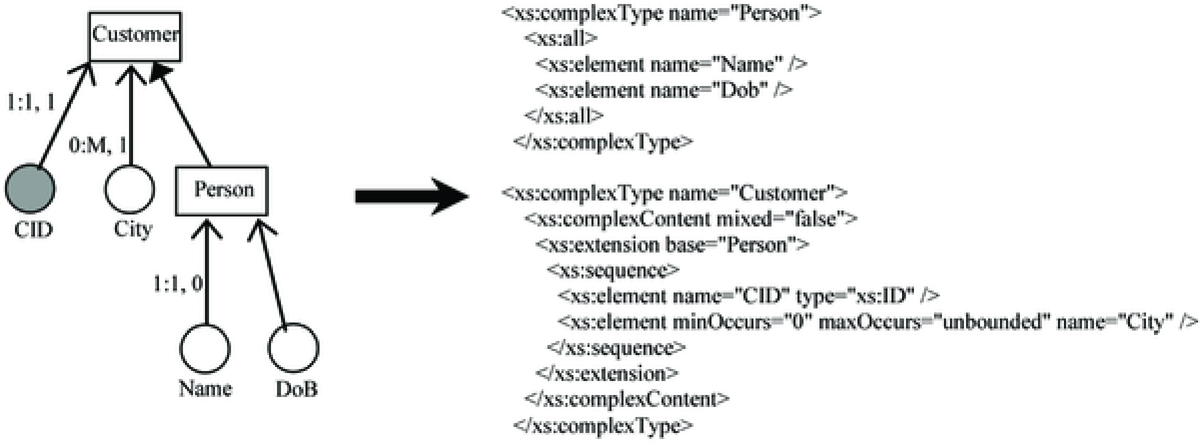
A conceptual model of semi-structured data deals with high level representation of the candidate application domain in order to capture the user ideas using rich set of semantic constructs and interrelationship thereof. Besides some similar characteristics of structured (classical) database system, several crucial characteristics are added complexity for the design of semi-structured database system. For effective design of such system, the intended conceptual model must be capable to adopt the rapidly data evolving characteristics, representation of irregular and heterogeneous structure, hierarchical relations along with the non – hierarchical relationship types, cardinality, n – array relation, ordering, representation of mixed content etc. (Necasky, 2006). Examples of Semi-structured data:

- Forms
- Classified ads
- Jobs postings
- Tax reports
- XML documents
- Emails

Figure 14 exemplify how to expose many business opportunities for leveraging semi-structured data to help extract business value and visual representation of the relationships between data elements to build useful systems based on these opportunities:

- Call center transcripts
- Technician repair notes

- Medical history
- Billing history
- Loan applications



```
<xs:complexType name="Person">
  <xs:all>
    <xs:element name="Name" />
    <xs:element name="Dob" />
  </xs:all>
</xs:complexType>

<xs:complexType name="Customer">
  <xs:complexContent mixed="false">
    <xs:extension base="Person">
      <xs:sequence>
        <xs:element name="CID" type="xs:ID" />
        <xs:element minOccurs="0" maxOccurs="unbounded" name="City" />
      </xs:sequence>
    </xs:extension>
  </xs:complexContent>
</xs:complexType>
```

Figure 14. Representation of Relationships

3.8 Preparing Clinical Trials Data Ontology

Key data elements fields that should be included in each clinical trial study. These data elements is called nodes in the graphical databases landscape because they are considered key to the clinical study.

Table 4.

The clinical trials data ontology (CTDO)

Data Elements	Definition
Sponsor	The name of primary organization that oversees implementation of study and is responsible for data analysis

Table 4.

Cont.

Collaborators	Providing support, including funding, design, implementation, data analysis and reporting. The data provider is responsible for confirming all collaborators before listing them. Provide up to 10 full names of collaborating organizations.
Oversight	Indicate whether this trial includes an intervention subject to US Food and Drug Administration (FDA)
Protocol Outcome	The written description of a clinical study. It includes the study's objectives, design, and methods. It may also include relevant scientific background and statistical information.
Enrollment	The number of participants in a clinical study. The "estimated enrollment" is the number of participants that the researchers need for the study.
Arm Group	A group or subgroup of participants in a clinical trial that receives specific interventions, or no intervention, according to the study protocol. This is decided before the trial begins.
Intervention	A process or action that is the focus of a clinical study. This can include giving participants drugs, medical devices, procedures, vaccines, and other products that are either investigational or already available. Interventions can also include noninvasive approaches such as surveys, education, and interviews.
Text Block	Include the text that describe the intervention

Table 4.

Cont.

Eligibility	Summary criteria for participant selection. The preferred format includes lists of inclusion and exclusion criteria
Contact	Person providing centralized, coordinated recruitment information for the entire study.
Investigator	A researcher involved in a clinical study. Related terms include Site Principal Investigator, Site Sub-Investigator, Study Chair, Study Director, and Study Principal Investigator.
Address	Mailing address for the board, including street address, city, state or province, postal code, and country
Facility	Full name of the organization where the protocol is being conducted
Location/country	Location of the facility or organization that carrying out the clinical trial study
Links	A Web site directly relevant to the protocol may be entered, if desired. Do not include sites whose primary goal is to advertise or sell commercial products or services
References	Citations to publications related to the protocol: background and/or results. Provide either the PubMed Unique Identifier (PMID)
Responsible Party	1. the sponsor of the clinical trial or 2. the principal investigator of such clinical trial
Browse	Browse the content of a clinical trial

Table 4.

Cont.

Group	For observational studies specify the predefined participant groups (cohorts) to be studied, corresponding to Number of Groups specified under Study Design
Milestones	Definition: Any specific events or time points in the trial when the numbers of participants are reported may be added
Period	Discrete stages of a clinical trial during which numbers of participants at specific significant events or points of time are reported
Participants Flow	Progress of research participants through each stage of a trial in a tabular format, including the number of participants who dropped out of the clinical trial.
Measure Category	Name of distinct category for a baseline measure, if reporting categorical data.
Measure	Name and description of a characteristic measured at the beginning of the trial
Baseline	A table of demographic and baseline data for the entire trial population and for each arm or comparison group
Analysis	Explanation of how the number of participants for analysis was determined
Results outcome	trial results after the trial is completed

Table 4.

Cont.

Vocabulary Term	Default value for Source Vocabulary Name to be applied to all adverse event terms
Certain Agreements	Information certifying whether there exists an agreement between the sponsor or its agent and the principal investigators
Point of contact	for scientific information about the posted clinical trial results
Clinical Study	A research study using human subjects to evaluate biomedical or health-related outcomes. Two types of clinical studies are Interventional Studies and Observational Studies.

3.9. Limitations of Neo4j database:

In this section the limitations of the tool used in this research study of this research (Neo4j) is listed to be the direction that the scientists have to solve and deal with. These limitations are:

- Add a data cleaning and data mining capabilities to the architecture of this research
- The concept of graph databases is a new discipline to use in clinical informatics systems
- Truthful recourses needed for installation and configuration
- Literature review articles
- Most of the applications that use Neo4j are used in small number of areas (Logistics, social media, recommendation systems, and master data management).
- Thus Neo4j is flexibility of not having a defined database schema can cause problems that could hinder the efficiency of the query results.

The graphical database Neo4j necessitate high advanced technologies in terms of knowledge in the tool, and hardware requirements to function properly and perform up to expectation.

Graph databases software is immature compared to the traditional RDBMS. The free versions available online don’t provide the comprehensive capabilities for full scale projects. Moreover, the online accessible platforms such as Graph Gist, Graph ETL, and Neo4j Community edition do not provide a complete set of big data components.

In addition, Neo4j software installation instructions are not direct specifically on Windows OS or Mac OS as there isn’t an executable installer for these tools. For example, Neo4j has many versions. Each version requires a set of instructions and compatibility.

In conclusion also, Neo4j database can’t partitioned, this suggest a dilemma to address; that will end up with a whole dataset in one server. This situation will put Neo4j scalability on test as to overcome this obstacle to scale vertically every time a space added on the server for more data.

CHAPTER 4

RESEARCH FINDINGS

The research aims to deliver a conceptual solution to a very specific problem instance in the area of clinical information retrieval, accompanied with new graph database solution to yield unique features that are not available in traditional relational database management systems (RDBMS).

4.1 Analysis

The research question and problem of this thesis is important to the medical and healthcare industries. The research question focuses on the integration of search engine, information extraction, information retrieval, and data management to determine the best treatment for a certain disease. The research problem focuses on the development and implementation of a Graphical Database Architecture for Clinical Trials.

4.2 Theoretical Solution

By applying the mechanisms of the research design, the study of this research is required to construct an initial theoretical scenario related to the subject of research the clinical trials (CT). This theoretical scenario provides an adequate scheme for the research.

This hints to a theoretical story about the events, structures and phases occur during a clinical trial. The outcome is a path in determining what data to collect and the approaches to analyze the data. This theoretical model of clinical trials (CT) scenarios provides a distinctive knowledge that can be used to improve application of analyzing and mining the clinical trials (CT) data. Our theoretical model emphases on assertion of achieving the goal of the study of this research in terms of outcome and efficiency.

The theoretical model in Figure 15 below demonstrates the routine scenario of clinical trials (CT) process. At the beginning clinical trials (CT) is a crucial link between the research and development (R&D) chain in the health sector. There are three crucial element must achieved in every clinical trial study, these elements are: Quality, Safety, and Efficacy.

The starting point of the clinical trials (CT) process is when the scientists or researchers begin to analyze the disease and investigate a possible treatment. Preclinical trials then establish initial safety and effectiveness before testing on human beings. These tests are often done in the laboratory, using test tubes research. After the preclinical trials, the next step is to check for safety (Phase I) by investigating treatment safety and research how it works and behaves in the human body. Usually this process takes between weeks and months and involve less than 100 participants. Phase II includes testing the efficacy of the treatment on hundreds of participants who have the disease, this phase is important to investigate side effects and risks. The timeframe of (Phase II) takes sometime up to several years. The next step (Phase III) is a very important step before seeking final approval. At this point the study pursues to establish benefit-risk, the right patients, and the best way to manage side effects of the treatment. This time the tests will be conducted on 1000’s of participants to make sure that all aspects of the study are follow the regulatory constraints.

Now the treatment results are confirmed and met the safety, quality, and efficacy measures. The governing body or regulators (in USA case Food and Drug Administration FDA is the regulator) after reviewing all the previous phases grant their approval and authorize the treatment/drug for use.

In Phase IV studies are concluded after the drug or treatment has been marketed to collect information on the drug's effect in numerous patients and any side effects associated with use.

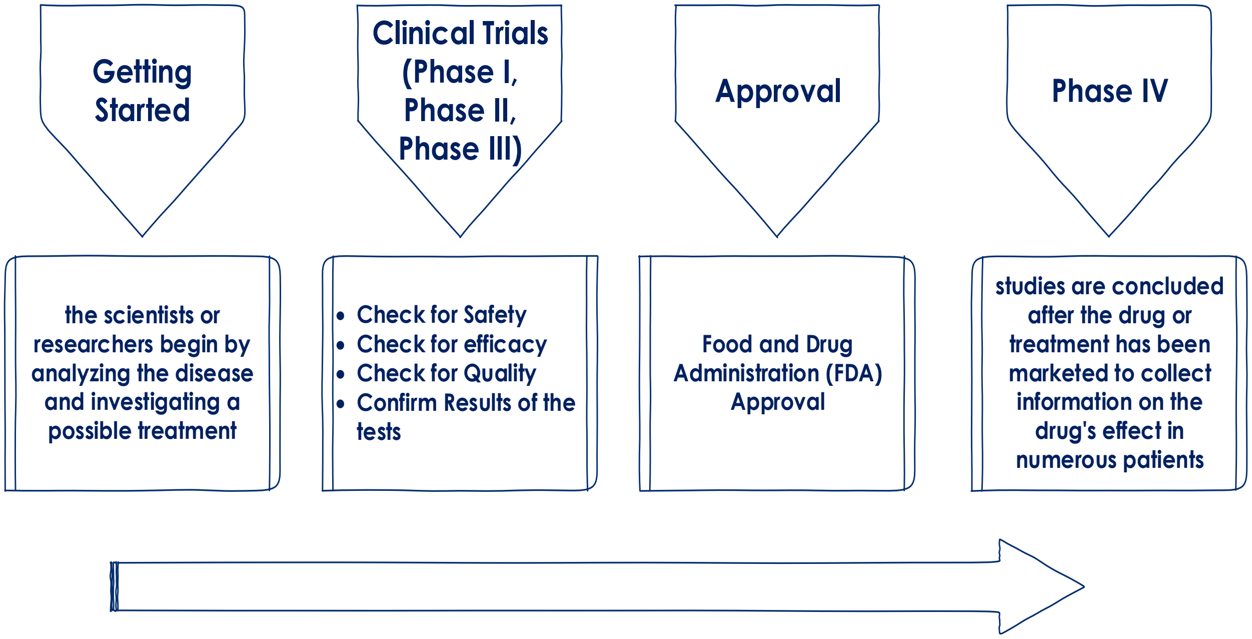

Getting Started
Clinical Trials (Phase I, Phase II, Phase III)
Approval
Phase IV
the scientists or researchers begin by analyzing the disease and investigating a possible treatment
• Check for Safety
• Check for efficacy
• Check for Quality
• Confirm Results of the tests
Food and Drug Administration (FDA) Approval
studies are concluded after the drug or treatment has been marketed to collect information on the drug's effect in numerous patients

Figure 15. Visual Scenario of a Clinical Trial (CT)

4.3 Visual Representation of Methodology Findings

Base on the data elements extracted in Chapter 3, a visual representation for the Clinical Trial is developed. This visual depiction going to serve as a conceptual roadmap to incorporate all these data elements in the Graphical Database Architecture (GDA) for Clinical Trials (CT). Six main "graphical" groups in our visual representation are identified to constitute Clinical Trial Ontology (CTO):

4.3.1 First Group: <u>Study Type:</u>

This data element describes nature of the investigation:

Interventional or clinical study: the starting point of the study illustrated in Figure 16 when individuals are assigned by an investigator based on a protocol to receive specific interventions (diagnostic, therapeutic…etc.).

Observational study: where biomedical results are assessed in pre-defined groups of individuals. Subjects in the study may receive diagnostic, therapeutic, or other interventions. But the key difference is the investigator didn't assign a precise intervention for the study.

Expanded Access: Expanded Access records are used to index all types of non-protocol access to experimental treatments, including protocol exception, single-patient IND (Investigational New Drug), treatment IND, compassionate use, emergency use, continued access and parallel track.

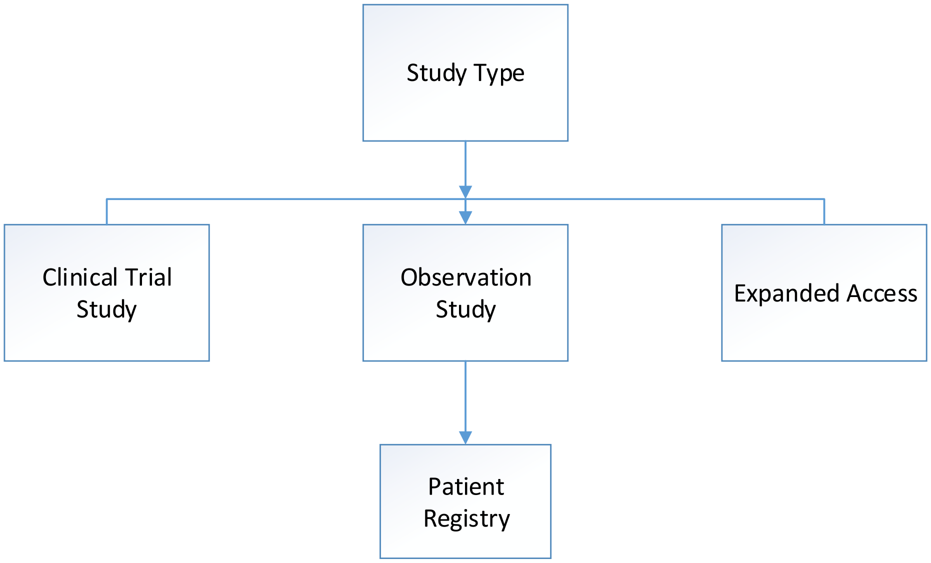


Figure 16. Study Type Label in the Clinical Trial Ontology (CTO)

4.3.2 Second Group. **<u>Recruitment:</u>** All Studies: showed in Figure 17 Regardless if they are recruiting new participants or not

Open Studies: Studies that are currently recruiting participants, will be recruiting participants in the future, or involve drugs that are available for expanded access. Recruitment statuses for open studies are:

- Recruiting

- Not yet recruiting
- Available for expanded access

Closed Studies: Clinical studies that are no longer recruiting participants for various reasons (i.e. the study is completed). Recruitment statuses for closed studies are:

- Study results
- Study by topic
- Locations
- Secondary data elements

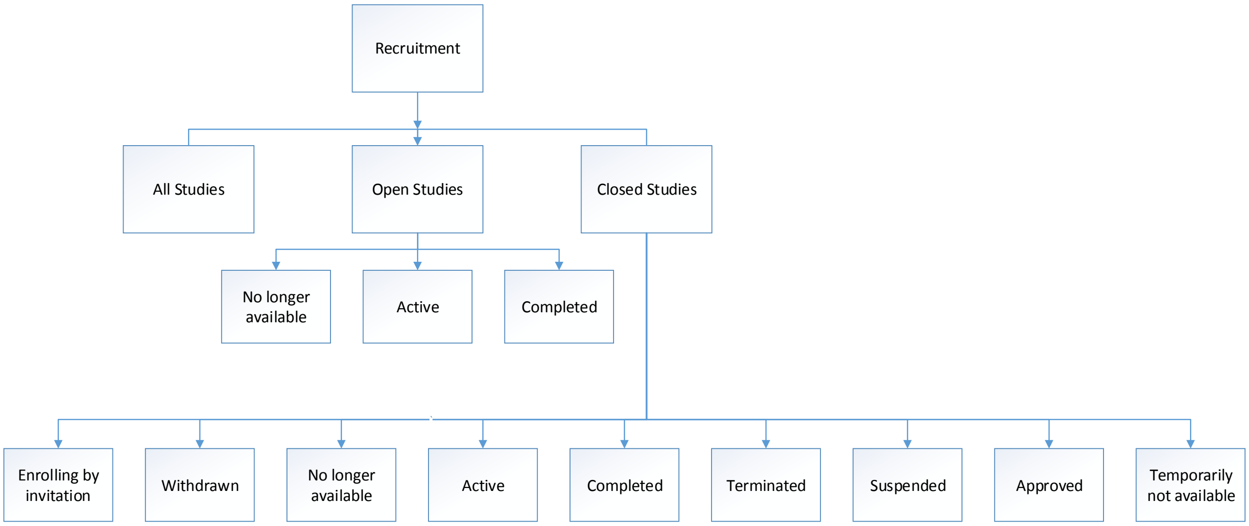


Figure 17. Recruitment Label in the Clinical Trial Ontology (CTO)

4.3.3 Third Group. **Study Results:** this label for clinical study results that were posted in the ClinicalTrials.gov results database. Figure 18 explain three main types:

- All Studies
- Studies with Results
- Studies without Results

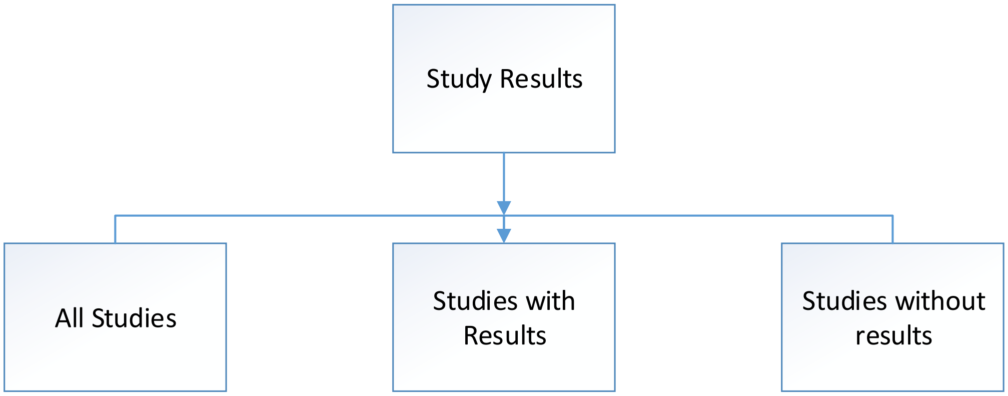


Figure 18. Study Result Label in the Clinical Trial Ontology (CTO)

4.3.4 Fourth Group. **Study by topic:** In *Figure 19* illustrate the following:

Conditions: describe a specific disease, disorder, syndrome, illness, or injury that is being studied.

Interventions: describe the process or action that is the focus of a clinical study. This can include giving participants drugs, medical devices, procedures, and vaccines.

Title Acronym: the study title acronym or an exact phrase from the brief or official title of the study.

Outcome Measures: a measurement decided on in advance that is used to determine the effect of interventions on participants in a clinical trial.

Sponsor and Collaborators: this specify the Sponsor (Lead) or Collaborator. A sponsor is the organization or person (a Sponsor-Investigator) who oversees the clinical study and is responsible for analyzing the study data. While a collaborator is an organization other than the sponsor that provides support for a clinical study.

Sponsor (Lead): Use this field to specify the Sponsor (Lead). A sponsor is the organization or person (a Sponsor-Investigator) who oversees the clinical study and is responsible for analyzing the study data

Study IDs: a clinical study by an identification number. Every clinical study is associated with identification numbers assigned to the clinical study protocol by the sponsor, funders. ClinicalTrials.gov assigns a ClinicalTrials.gov Identifier, or NCT Number, a unique identification code given to each clinical study registered on ClinicalTrials.gov. The format is the letters "NCT" followed by an 8-digit number (for example, NCT00000568).

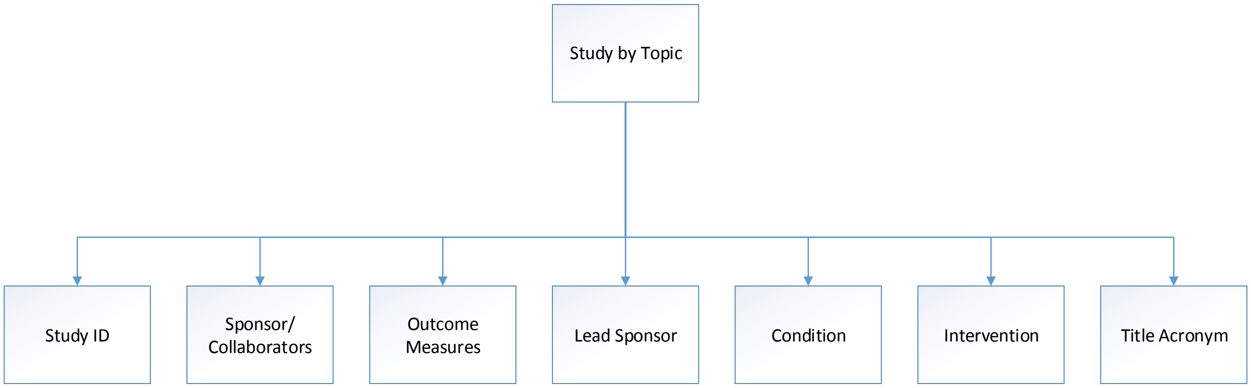


Figure 19. Study by Topic Label in the Clinical Trial Ontology (CTO)

4.3.5 Fifth Group. Locations exemplified in Figure 20:

Location Terms: to identify a specific city (such as New York) or facility (such as the Mayo Clinic), enter it as a location term.

Country/State: Studies are often conducted in many locations around the world.

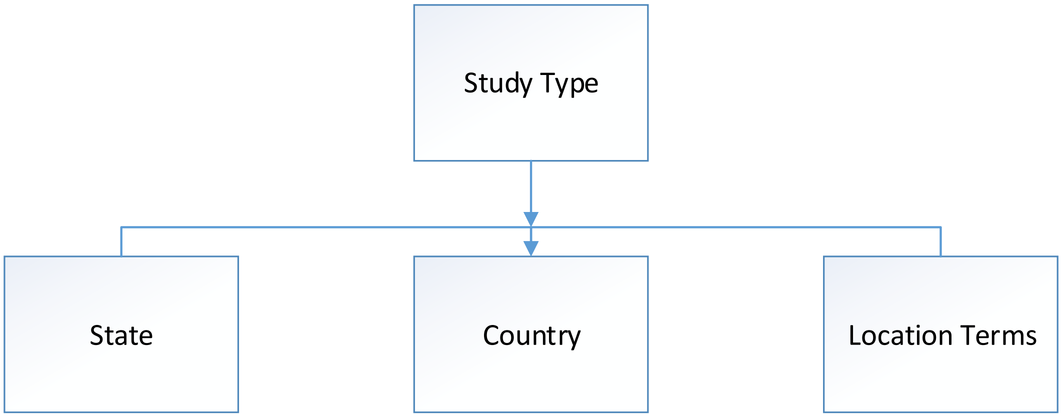


Figure 20. Study type label in the clinical trial ontology (CTO)

4.3.6 Sixth Group. Secondary Criteria visualized in Figure 21:

- Gender
- Age
- Clinical Trial Phase
- Funding Type
- Safety measurement
- Clinical trial adding date: is the date that summary clinical study protocol information was first submitted to the registry
- Clinical trial update date: is the most recent date that changes to study information were submitted to the registry

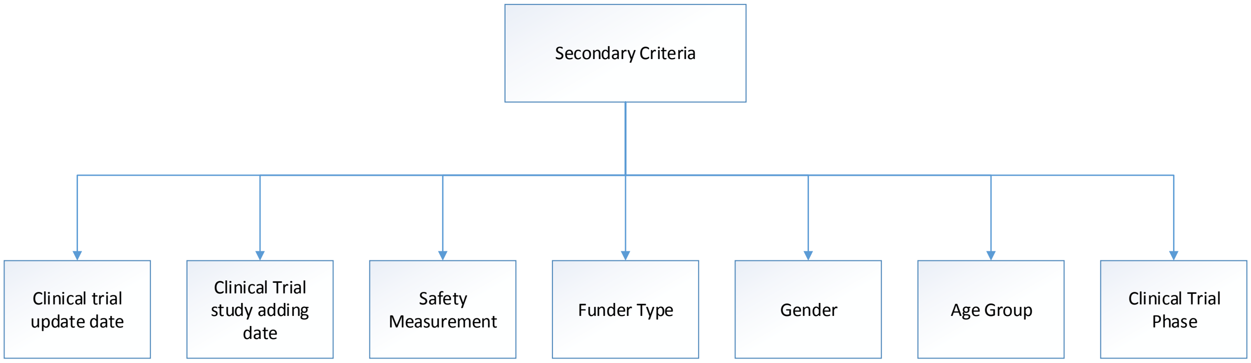


Figure 21. Secondary criteria labels in the clinical trial ontology (CTO)

4.4 Setting GDA for Clinical Trials Environment

This section describes the setup of the required software for creating and running the GDA. This section contains the following:

As discussed earlier in the research, Neo4j offers flexibility of two modes to start Neo4j.

- An embedded database in a Java application
- A server mode that facilitate REST

As discussed in the research Neo4j is coded primarily in Java and leverages a custom storage format and the facilities of the Java Transaction Architecture (JTA) to provide XA transactions. The Java application program interface (API) coded in an object-oriented way of working with the nodes and relationships of the graph.

4.4.1 Embedded Database in a Java Application. In embedded mode, Neo4j is used by any client code that’s capable of running within a Java Virtual Machine (JVM). The advantage is the embedded mode can be used directly with Java clients As Figure 22 shows, which make direct connection with the core Java Neo4j libraries. The other way is by adding an additional language-specific (i.e. JavaScript, Ruby) and frameworks provided for most JVM-based languages.

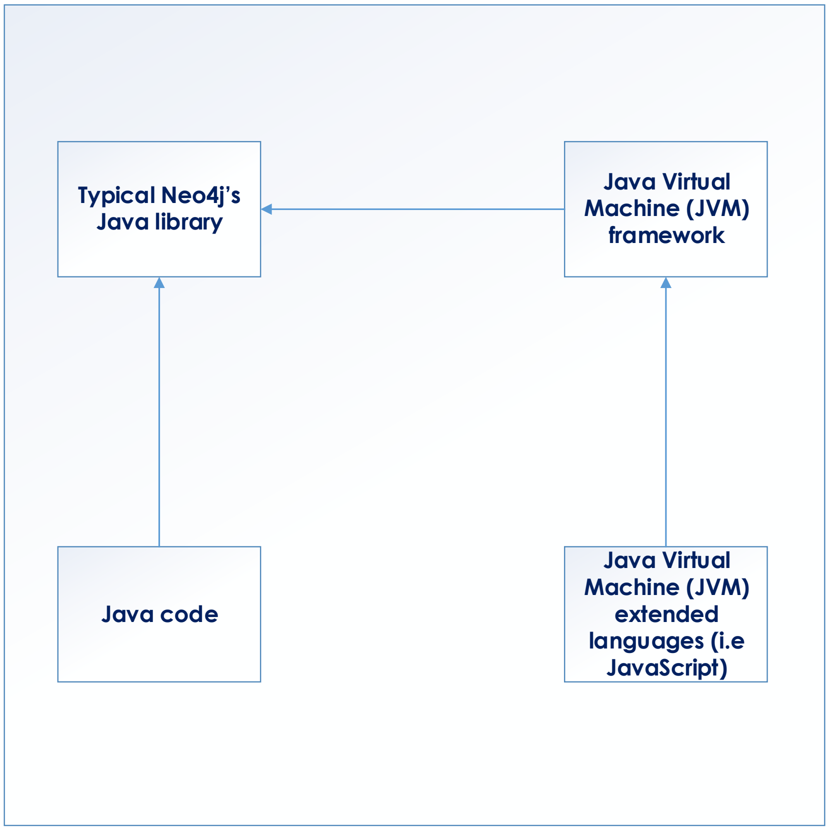

Typical Neo4j's Java library
Java Virtual Machine (JVM) framework
Java code
Java Virtual Machine (JVM) extended languages (i.e JavaScript)

Figure 22. The Flow of Embedded Neo4j Server

Embedded mode requires JAR files to be bundled or made available to the application when it starts up. Then the application is going to gain access to the Neo4j database by executing an instance of the *GraphDatabaseService* interface. Afterwards the application use this reference to interact with Neo4j, using all of the available APIs. While *EmbeddedGraphDatabase* class usually is used for a single machine setup; *theHighlyAvailableGraphDatabase* class is used for a multiple application setup.

4.4.2 Embedded Graphical Database. First of all, to create a database, a *GraphDatabaseFactory* class is needed,

```
GraphDatabaseFactory graphDbFactory = new GraphDatabaseFactory();
```

Then

```
GraphDatabaseService graphDb = graphDbFactory     .newEmbeddedDatabase("data/CTdb");
```

4.4.3 Server Mode. In server mode, the client code work together with Neo4j's server through the HTTP protocol that includes the REST API. First REST API is a web services are purpose-built web servers that support the needs of a site or any other application. Client programs use application programming interfaces (APIs) to communicate with web services. An API exposes a set of data and functions to facilitate interactions between computer programs and allow them to exchange information. As depicted in Figure 23, a Web API is the face of a web service, directly listening and responding to client requests (Masse, 2011).

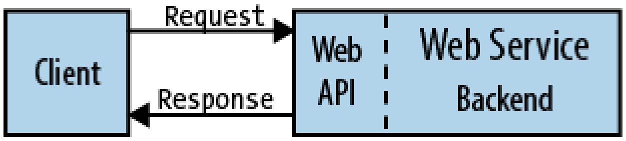


Figure 23. REST Web API

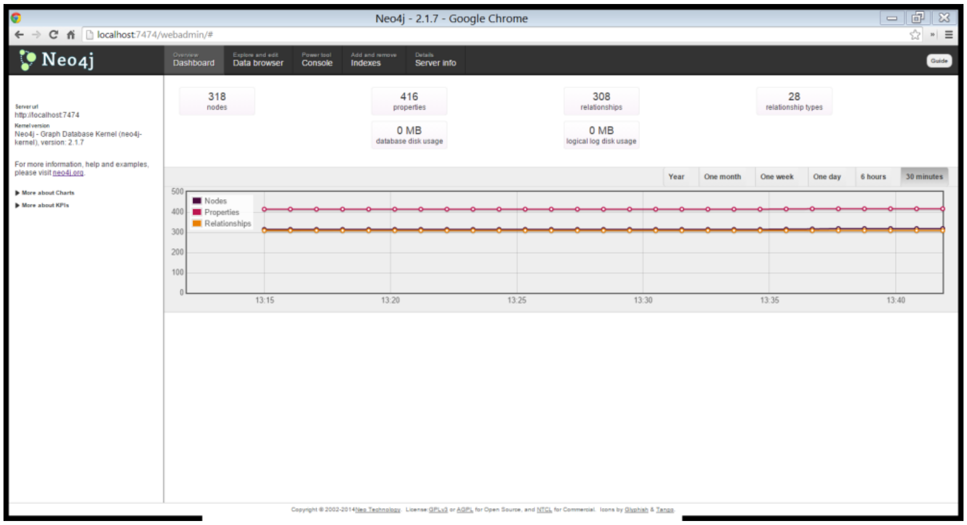
Neo4j server mode offers the option to extend REST functionality when required. The REST API can be used with a HTTP-enabled client, and another option is using one of the remote REST client APIs available for a variety of different languages and frameworks.

4.5 Demonstration of Concept

4.5.1 Graphical Database Architecture (GDA)

- GDA's Neo4j Dashboard
- GDA Data browser
- GDA Power tool console
- Index console
- GDA's Neo4j embedded server Information details

4.5.1.1 Neo4j Dashboard for GDA. The Dashboard in Figure 24. The Main Dashboard of Neo4j delivers a quick monitoring of database content and activity, convenient summary of Key Performance Indicators (KPI) estimate of node, relationship, indexes and property numbers.



Server url
http://localhost:7474
Kernel version
Neo4j - Graph Database Kernel (neo4j-kernel), version: 2.1.7

For more information, help and examples, please visit neo4j.org.

318 nodes

416 properties

308 relationships

28 relationship types

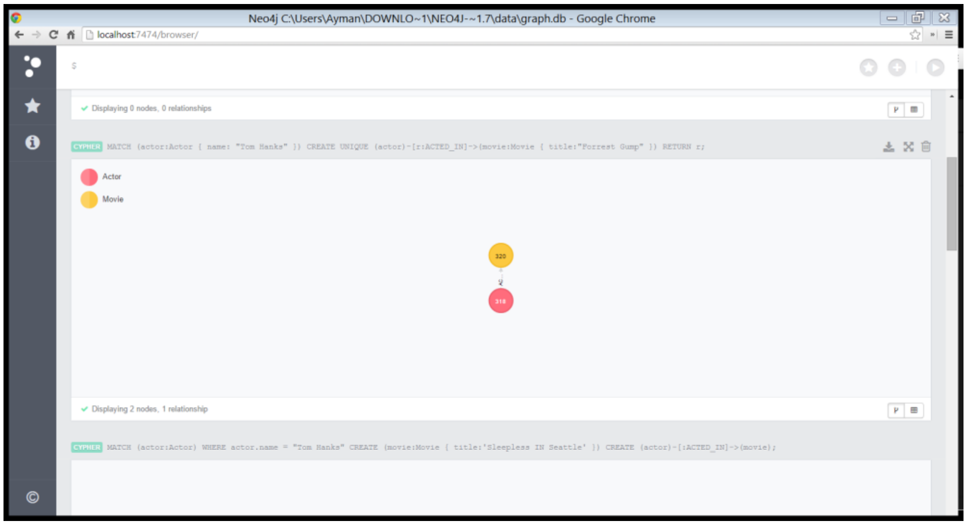
0 MB database disk usage

0 MB logical log disk usage

Copyright © 2002-2014Neo Technology. License:GPLv3 or AGPL for Open Source, and NTCL for Commercial. Icons by Glyphish & Tango.

Figure 24. The Main Dashboard of Neo4j

4.5.1.2 GDA Data browser. The Data browser tab provides two views for exploring the graph. The default inspection view displays details about nodes, relationships and property values. The data browser in Figure 25 is a convenient tool for easy data creation and manipulation. Also anther Data browser function is the ability to view is a simple graph visualization.


Neo4j C:\Users\Ayman\DOWNLO~1\NEO4J-~1.7\data\graph.db - Google Chrome
localhost:7474/browser/
$
Actor
Movie
320
318
CYPHER
CYPHER

✓ Displaying 0 nodes, 0 relationships

```
MATCH (actor:Actor { name: "Tom Hanks" }) CREATE UNIQUE (actor)-[r:ACTED_IN]->(movie:Movie { title:"Forrest Gump" }) RETURN r;
```

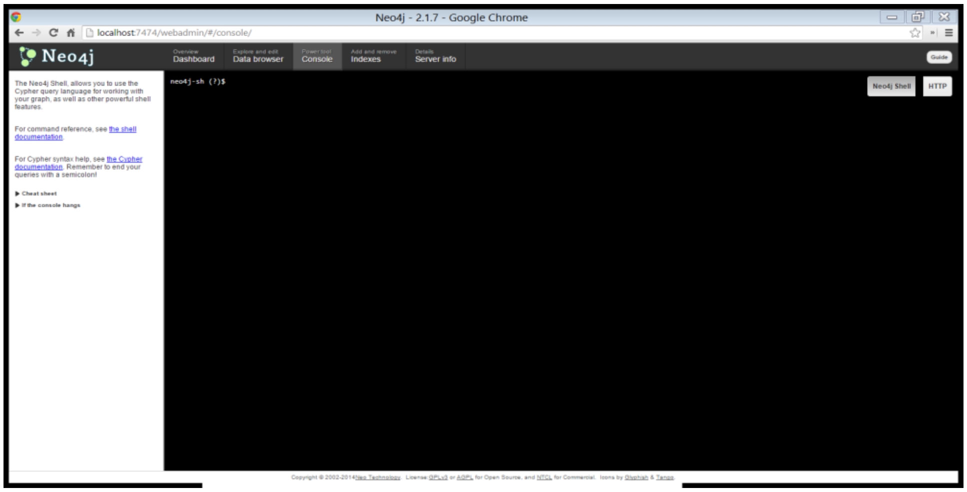
✓ Displaying 2 nodes, 1 relationship

```
MATCH (actor:Actor) WHERE actor.name = "Tom Hanks" CREATE (movie:Movie { title:'Sleepless IN Seattle' }) CREATE (actor)-[:ACTED_IN]->(movie);
```

Figure 25. Neo4j’s Data Browser for GDA

- Enter a Cypher query to view results
- Inspect nodes, relationships and property values
- Node and relationship results are clickable buttons
    - Enter a Cypher query to view results
    - Inspect nodes, relationships and property values
    - Node and relationship results are clickable buttons
    - Enter a Cypher query to view results
    - Inspect nodes, relationships and property values
    - Node and relationship results are clickable buttons
    - Enter a Cypher query to view results
    - Inspect nodes, relationships and property values
    - Node and relationship results are clickable buttons
    - Enter a Cypher query to view results
    - Inspect nodes, relationships and property values
    - Node and relationship results are clickable buttons

4.5.1.3 GDA Power tool console. The shell illustrate in Figure 26 is a command-line shell for running Cypher queries. There’s also commands to retrieve information about the database.



The Neo4j Shell, allows you to use the Cypher query language for working with your graph, as well as other powerful shell features.

For command reference, see the shell documentation.

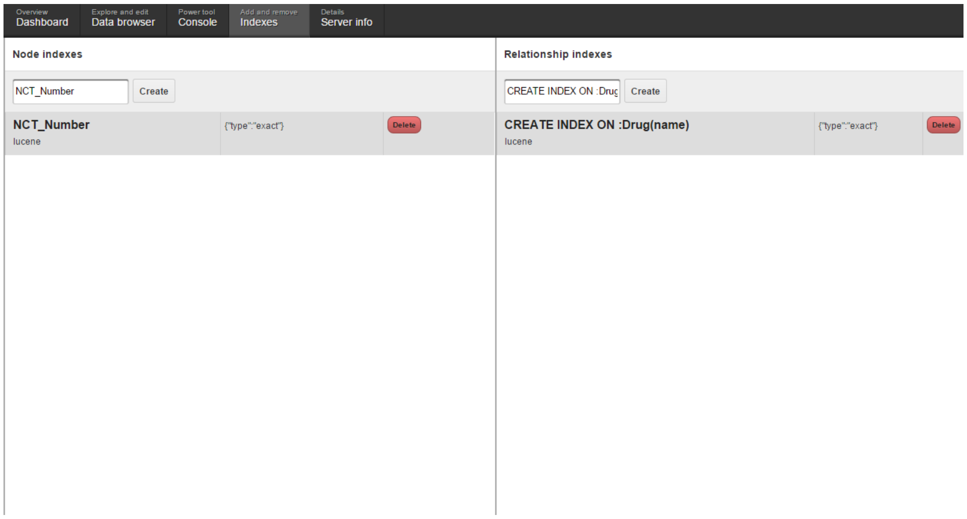
For Cypher syntax help, see the Cypher documentation. Remember to end your queries with a semicolon!

```
neo4j-sh (?)$
```

Copyright © 2002-2014Neo Technology. License:GPLv3 or AGPL for Open Source, and NTCL for Commercial. Icons by Glyphish & Tango.

Figure 26. Neo4j Power Tool Console

4.5.1.4 Index console. This interface illustrated in Figure 27 create and remove indexes from database. As discussed earlier, Neo4j has two types of indexes, node and relationship indexes. With node indexes, a specific nodes (i.e. Clinical Trial’s National identification number NCT) can be found, and relationship indexes is to create and archive the relationships of these nodes. Although the index console is not capable of creating more complex structures for big enterprise datasets which require to facilitate Neo4j REST client (REST API from Java).



Node indexes

NCT_Number

NCT_Number
lucene
{"type":"exact"}

Relationship indexes

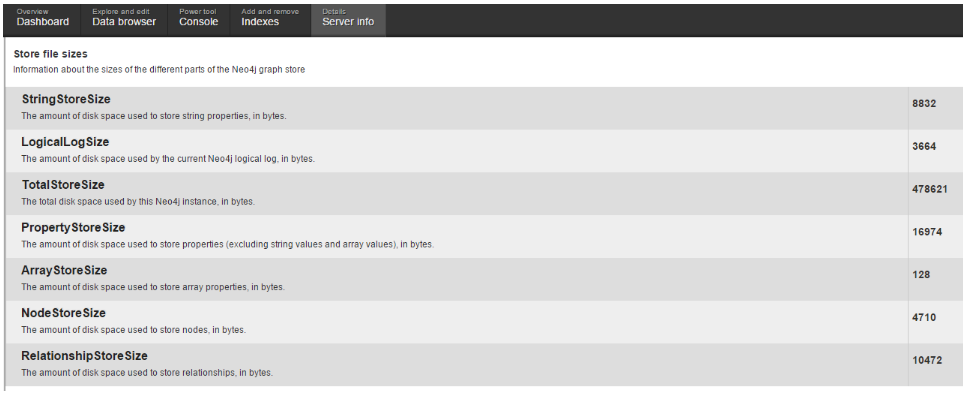
CREATE INDEX ON :Drug

CREATE INDEX ON :Drug(name)
lucene
{"type":"exact"}

Figure 27. Neo4j Indexing tool for GDA

4.5.1.5 Neo4j Embedded Server. This tool is important to maintain and manage the Neo4j database backend. It provides several important information, for example in Figure 28 Information about the sizes of the different parts of the Neo4j graph store:

- Neo4j cache, transactions, store file sizes, configuration, and diagnostic of the DB status.
- Information about the interface of MBean (included in Java Platform)
- Java logging record
- Java implementation information



Store file sizes
Information about the sizes of the different parts of the Neo4j graph store

StringStoreSize The amount of disk space used to store string properties, in bytes.	8832
LogicalLogSize The amount of disk space used by the current Neo4j logical log, in bytes.	3664
TotalStoreSize The total disk space used by this Neo4j instance, in bytes.	478621
PropertyStoreSize The amount of disk space used to store properties (excluding string values and array values), in bytes.	16974
ArrayStoreSize The amount of disk space used to store array properties, in bytes.	128
NodeStoreSize The amount of disk space used to store nodes, in bytes.	4710
RelationshipStoreSize The amount of disk space used to store relationships, in bytes.	10472

Figure 28. GDA’s Neo4j Embedded Server Information

CHAPTER 5

CONCLUSION & FUTURE RESEARCH

This thesis has laid the groundwork for a graphical database platform for clinical trials (CT) data and has leveraged it to investigate multiple useful and high impact aspects of graph theory. The thesis discusses the strengths and weaknesses of the graph databases, specifically Neo4j. Also in this chapter, the researcher provides the recommendations of the study of this research. Graph databases represents a truly significant structures. The technology aspect of graph databases especially (Neo4j) is here, open and available to all in the form of the modern graph database.

Graph databases algorithms and analytical techniques are sophisticated but not difficult to use, the need is to comprehend how to apply these new technologies to attain any research objectives. Modeling with graphs, graph database architecture, designing and implementing a graph database solution, and applying graph algorithms to complex business problems is the way to build a unique information systems.

5.1 Contribution Summary

This thesis adds to the field of health informatics, database management and big data by applying an innovative model and new way of handling semi-structured data. The new concept try to introduce a dynamic graphical database model that provides efficiency, high availability, handling huge number of nodes (32 billion), edges (32 billion), and up to 64 billion properties. As mentioned earlier that will allows all stakeholders that may be involved or interested in clinical trials data to utilize it for better data management and faster processing time.

5.2 Limitation and Future Work

The graph database provides a strong but novel data modeling technique even though it does not in itself provide adequate reasoning for replacing a long-standing, well-understood data platform; there must also be an immediate and very important practical benefit. The research established that Neo4j is still an immature tool, hence it is growing progressively to be able to offer capabilities to serve big enterprise projects and systems.

The current work is still in the process of implementing the concept of GDA for clinical trials. One of the aspects that will be considered is to extend the architecture into separate server not only a server embedded within the Java environment. The current architecture though represent the first step towards a comprehensive architectural model that can imitate for example SOA. There are many options to extend the GDA concept:

First option is the integration between Neo4j server with Neo4j and .NET Development Environment that includes a Neo4jClient API and developing a .NET and Neo4j web application. The second option is pairing Python programming language with Neo4j in a development environment that includes: Integrated development environment (IDE) like Eclipse, alongside Python language driver plugin for Neo4j. The third option is using Java with Neo4j. The Java integrates Neo4j server instance with the Neo4j JDBC driver, also known as Neo4j JDBC or simply NJDBC.

Finally measuring the performance of the GDA for clinical trials (CT) frequently is crucial to determine its excellence and productivity due to the continuous growth of clinical data. The amount of data sets of the new clinical trials, studies, and investigation may affect the performance of GDA.

References

Abrahiem, R., (2007). Designing and Implementing Prototype Models for an Integration of Near-Real-Time Data Warehousing Architecture with Service-Oriented Architecture. Electro/Information Technology, 2007 IEEE International Conference

Agneeswaran, Vijay Srinivas. Big Data Analytics Beyond Hadoop. Print.

Angles, R., "A Comparison of Current Graph Database Models," Data Engineering Workshops (ICDEW), 2012 IEEE 28th International Conference on , vol., no., pp.171,177, 1-5 April 2012 doi: 10.1109/ICDEW.2012.31

Anton, A. & Carter, R. & Earp, J. & Williams, L. (2001). EPRAM: Evolutionary Prototyping Risk Analysis & Mitigation (e-Commerce Software Development Process Document) NCSU Software Requirements Engineering Lab

Balasubramanian, Harish, "Performance Analysis Of Scalable Sql And Nosql Databases : A Quantitative Approach" (2014). Wayne State University Theses. Paper 327.

Buerck J, Feig D. Knowledge discovery and dissemination: a curriculum model for informatics. SIGCSE Bulletin. 2006; 38(4):48-51.

Castellano, M. & Mastronardi, G. & Tarricone, G. A Web Text Mining Flexible Architecture. World Academy of Science, Engineering and Technology 32 2007

Celko, Joe. Joe Celko's Complete Guide To Nosql. Waltham, MA: Morgan Kaufmann, 2013. Print.

Creswell, J. (2003). Research Design: Qualitative, Quantitative, and Mixed Methods Approaches. SAGE Publications International Educational and Professional Publisher, Thousand Oaks, London, New Delhi

Chhanda Ray Distributed Database Systems by Publisher: Pearson, Pearson Education India Release Date: June 2009

Clinicaltrials.gov,. 'Home - Clinicaltrials.Gov'. N.p., 2015. Web. 28 Feb. 2015.

Cullen, A. & Cecere, M. & Symons, C. & Cameron, B. & Cardin, L. & Orlov, L. & Belanger, B. (2007). The IT Strategic Plan Step-By-Step Deliver An Actionable Plan In A Reasonable Timeframe. Forrester Research, Inc.

El-Bathy, N.; Gloster, C.; Azar, G.; El-Bathy, M.; Stein, G., "Intelligent optimization models for disease diagnosis using a service-oriented architecture and management science," *Electro/Information Technology (EIT), 2012 IEEE International Conference on* , vol., no., pp.1,8, 6-8 May 2012

El-Bathy, N. & Chang, P. & Azar, G. & Abrahiem, R., (2010). An Intelligent Search of Lifecycle Architecture for Modern Publishing and Newspaper Industries Using SOA. IEEE, Electro/Information Technology, Normal, IL, United States, pp. 1-7

El-Bathy, N., & Azar, G., (2010) Intelligent Information Retrieval and Web Mining Architecture Applying Service-Oriented Architecture. KG. Saarbrücken, Germany: LAP LAMBERT Academic Publishing AG & Co.

Galin, D. (2004). Software Quality Assurance From theory to implementation. Pearson Addison Wesley

Gutierrez, A. & Serrano , A. (2008). Assessing strategic, tactical and operational alignment factors for SMEs: alignment across the organization’s value chain. International Journal of Value Chain Management 2008 - Vol. 2, No.1 pp. 33 – 56

Haynes, S. & Skattebo, L. & Singel, J. & Cohen, M. & Himelright, J. (2006). Collaborative Architecture Design and Evaluation. DIS 2006, June 26–28, 2006, University Park, Pennsylvania, USA. ACM 1-59593-341-7/06/0006

Hongcheng Huang; Ziyu Dong, "Research on architecture and query performance based on distributed graph database Neo4j," Consumer Electronics, Communications and Networks (CECNet), 2013 3rd International Conference on , vol., no., pp.533,536, 20-22 Nov. 2013

Healey, Bernard J, and Tina M Evans. Introduction To Health Care Services. Print.

Hurwitz, Judith et al. Big Data For Dummies. Hoboken, N.J.: Wiley, 2013. Print.

Jouili, S.; Vansteenberghe, V., "An Empirical Comparison of Graph Databases," Social Computing (SocialCom), 2013 International Conference on , vol., no., pp.708,715, 8-14 Sept. 2013 doi: 10.1109/SocialCom.2013.106

Jordan, Gregory. Practical Neo4j. Berkeley, CA: Apress, 2014. Print.

Lewis, C. & Poison, P. & Wharton, C. & Rieman, J. Testing a Walkthrough Methodology for Theory- Based Design of Walk-Up-and-Use Interfaces. In Proceedings of CHI 1990 Conference (Seattle, WA, Apr. 1-5). ACM, New York, 1990, pp. 235-242.

Lingli Fu; Sheng Ding; Tao Chen, "Clinical Data Management System," Biomedical Engineering and Computer Science (ICBECS), 2010 International Conference on , vol., no., pp.1,4, 23-25 April 2010

Lumsden, Joanna. Handbook Of Research On User Interface Design And Evaluation For Mobile Technology. Hershey, PA: Information Science Reference, 2008. Print.

Lui, Keith. Assessing Health Informatics And Computing Experiments. [S.l.]: Vdm Verlag Dr Mueller, 2010. Print.

Massé, Mark. REST API Design Rulebook. Sebastopol, CA: O'Reilly, 2012. Print.

McConnell, Steve. Rapid Development. Redmond, Wash.: Microsoft Press, 1996. Print.

Minelli, Michael, Michele Chambers, and Ambiga Dhiraj. Big Data, Big Analytics. [s.l.]: John Wiley & Sons Inc, 2013. Print.

Naser El-Bathy, Clay Gloster, Ghassan Azar, Intelligent Internet Search Technology Using a Novel Genetic Algorithm and a Service-Oriented Architecture, American Journal of Intelligent Systems, Vol. 3 No. 2, 2013, pp. 83-92. doi: 10.5923/j.ajis.20130302.05.

Neo4j Graph Database. 'Neo4j, The World's Leading Graph Database'. N.p., 2015. Web. 28 Feb. 2015.

Ohlhorst, Frank. Big Data Analytics. Hoboken, N.J.: John Wiley & Sons, 2013. Print.

Oracle.com,. 'Java SE - Downloads | Oracle Technology Network | Oracle'. N.p., 2015. Web. 28 Feb. 2015.

Papakonstantinou, Y.; Garcia-Molina, H.; Widom, J., "Object exchange across heterogeneous information sources," Data Engineering, 1995. Proceedings of the Eleventh International Conference on , vol., no., pp.251,260, 6-10 Mar 1995
doi: 10.1109/ICDE.1995.380386

Pollack, Mark. Spring Data. Sebastopol, CA: O'Reilly, 2012. Print.

Pinelle, D. & Gutwin, C. (2002). Groupware Walkthrough: Adding Context to Groupware Usability Evaluation. CHI 2002, April 20-25, 2002, Minneapolis, Minnesota, USA. Copyright 2002 ACM 1-58113-453-3/02/0004…$5.00

Panzarino, Onofrio. Learning Cypher. Birmingham, U.K.: Packt Pub., 2014. Print.

Robinson, Ian, James Webber, and Emil Eifrem. Graph Databases. Sebastopol, Calif.: O'Reilly Media, 2013. Print.

Sjobergh, J.; Kuwahara, M.; Tanaka, Y., "Visualizing Clinical Trial Data Using Pluggable Components," Information Visualisation (IV), 2012 16th International Conference on , vol., no., pp.291,296, 11-13 July 2012

Sadalage, Pramod J, and Martin Fowler. Nosql Distilled. Upper Saddle River, NJ: Addison-Wesley, 2013. Print.

Shneiderman, Ben. Sparks Of Innovation In Human-Computer Interaction. San Mateo, Calif.: Morgan Kaufmann Publishers, 1994. Print.

Shortliffe, Edward Hance, and James J Cimino. Biomedical Informatics. New York, NY: Springer, 2006. Print.

Sjöbergh, J.; Kuwahara, M.N.; Tanaka, Y., "Using Web-based Meme Media technologies to create an integrated visual environment for clinical trials," Frontier Computing. Theory, Technologies and Applications, 2010 IET International Conference on , vol., no., pp.366,371, 4-6 Aug. 2010

Singh, Hardeep, and Kulwant Kaur. Designing, Engineering, And Analyzing Reliable And Efficient Software. Hershey PA: Information Science Reference, 2013. Print.

Smith J, Buerck J. A multidisciplinary approach to graduate education in informatics - a proposal in process. Journal of Computing Sciences in Colleges. 2007;22(4):39-45.

T. Bray, J. Paoli, C.M. Sperberg-McQueen, E. Maler, F.Yergeau, and J. Cowan, editors, Extensible Markup Language (XML) 1.1 (Second Edition), http://www.w3.org/TR/2006/REC-xml11-20060816/, W3C Recommendation, 16 August 2006, edited in place 29 September 2006.

T. Catarci, T. Di Mascio, E. Franconi, G. Santucci, and S. Tessaris. An ontology based visual tool for query formulation support. In LNCS, volume 2889, pages 32--33. Springer, 2003.

Rowley, D. & Rhoades, D. (1992). The Cognitive Jogthrough: A Fast-Pace User Interface Evaluation Procedure. ACM 0-89791-51 3-5/92/0005-0389 1.50

Yunkai Liu; Vitolo, T.M., "Graph Data Warehouse: Steps to Integrating Graph Databases Into the Traditional Conceptual Structure of a Data Warehouse," Big Data (BigData Congress), 2013 IEEE International Congress on , vol., no., pp.433,434, June 27 2013-July 2 2013

Vaish, Gaurav. Getting Started With Nosql. Birmingham, UK: Packt Pub., 2013. Print.

Van Bruggen, Rik. Learning Neo4j. Birmingham, UK: Packt Pub., 2014. Print.

Watson, T. (2006). Data Management: Databases and Organizations. Wiley

Walls, Craig, and Ryan Breidenbach. Spring In Action. Greenwich [Conn.]: Manning, 2005. Print.

Widow, B., Hartenstein, R., & Hecht-Nielsen, R. (2005). 1917 Karl Steinbuch 2005, In IEEE Computational Intelligence Society Newsletter, August.

Wiegers, Karl Eugene, and Joy Beatty. Software Requirements. Redmond, Wash: Microsoft Press., 2013 Print.