FACILITIES MANAGEMENT KNOWLEDGE MAPPING FROM TEXT DOCUMENTS

A CASE STUDY FOR USING NLP FOR FACILITIES MANAGEMENT KNOWLEDGE MAPPING

A Thesis

by

TAO SONG

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Chair of Committee,	Sarel Lavy
Committee Members,	Eric Jing Du
	Boong Yeol Ryoo
	Youngre Noh
Head of Department,	Patrick Suermann

August 2018

Major Subject: Construction Management

Copyright 2018 Tao Song

ABSTRACT

Facility data are essential for any efficient facilities management and operations practices. However, due to the complexity and massive amount of information, interoperability is an important issue in facilities management. A knowledge map is a visual aid that can provide semantic clarity among different knowledge items and can lead directly to where the knowledge is stored. It can help facilities management staff save time when dealing with large amounts of complex data.

This thesis presents a way to create a knowledge map from unstructured text documents. It first uses a natural language processing technical method to recognize knowledge items and the relations among them from the facilities management documents; it then uses Protégé, an ontology-modeling tool, to create an Ontology Web Language based knowledge map to improve the interoperability, data management, and decision making.

The knowledge map was evaluated in terms of its accuracy, integrity, and readability. It was found that the relations in the map are accurate, and suggest two instances, equipment and manager, where the label content could be expanded to better reflect what a user would be looking for. The knowledge map represents facts that can be used by the inference engine and can be further developed as an expert system that can emulate the decision-making ability of a human expert.

ACKNOWLEDGEMENTS

I would like to take this opportunity to acknowledge the assistance, encouragement and support of the following people, without them I would not have been able to complete this thesis.

First of all, I would like to thank my thesis committee chair, Dr. Sarel Lavy, and members, Dr. Boong Yeol Ryoo, Dr. Eric Jing Du and Dr. Youngre Noh, for their time, expertise, and guidance, as well as every Construction Science faculty and staff member at Texas A&M University throughout the past two years.

I would also like to thank the members of facilities services at Texas A&M University for their help in my data collection and results evaluation processes. I really appreciate your knowledge, experience, and patience.

Furthermore, I would like to thank my classmates and other friends for all the assistance, wonderful memories and good times during my stay in Aggie land.

Finally, I would like to thank my parents and the rest of my family. It is their continuous support, encouragement and financial aid that made possible my study at Texas A&M University.

CONTRIBUTORS AND FUNDING SOURCES

This work was supervised by a thesis committee consisting of the committee chair, Dr. Sarel Lavy of the Department of Construction Science, and committee members Drs. Boong Yeol Ryoo and Jing Du of the Department of Construction Science and Youngre Noh of the Department of Landscape Architecture and Urban Planning.

The data used for this work were provided by the Property Management unit of Financial Management Operations of Texas A&M University.

All other work conducted for the thesis was completed independently by the student Tao Song, under the advisement of Dr. Sarel Lavy and the committee. The student did not receive any outside funding that contributed to the research or the compilation of this thesis.

NOMENCLATURE

AEC	Architecture, Engineering, and Construction
API	Application Programming Interface
BAS	Building Automation System
BIM	Building Information Modeling
COBie	Construction-to-Operations Building information exchange
FM	Facilities Management
FMS	Facilities Management System
FP	Frequent Pattern
IFC	Industry Foundation Classes
IFMA	International Facility Management Association
NIST	National Institute of Standards and Technology
NLP	Natural Language Processing
NLTK	Natural Language Toolkits
RDF/S	Resource Description Framework/Schema
OWL	Web Ontology Language
W3C	World Wide Web Consortium

TABLE OF CONTENTS

Page

LIST OF FIGURES

Page

LIST OF TABLES

Page

1. INTRODUCTION

Facilities management (FM) is a set of multi-disciplinary services and activities integrating people, place, process, and technology to ensure proper management of an asset during its life-cycle (Abdullah et al. 2013). Thus, FM has extensive information requirements (Becerik-Gerber et al. 2011). The use of innovative digital technologies and tools such as building information modeling (BIM), computer-aided FM systems, and building automation systems (BASs) has significantly improved the FM efficiency (Araszkiewicz 2017). However, different software and systems are characterized by high complexity in order to integrate heterogeneous devices that often come from a variety of vendors using different communication protocols (Tomašević et al. 2015). Although academic efforts have been made such as Construction Operations Building Information Exchange (COBie) and Industry Foundation Classes (IFCs), the uptake for interoperability in the industry is still shown to be slow. One reason for this is that these standards lack semantic clarity in mapping entities and relationships, which greatly hinders their application (Chen and Luo 2016). One way to define semantic clarity is using a knowledge map to map a dataset's unknown terms in known terms. The demand is met if the mapping between the data structure and the knowledge structure is straightforward, meaning that the meaning of each data element has to be clear (van Atteveldt 2001).

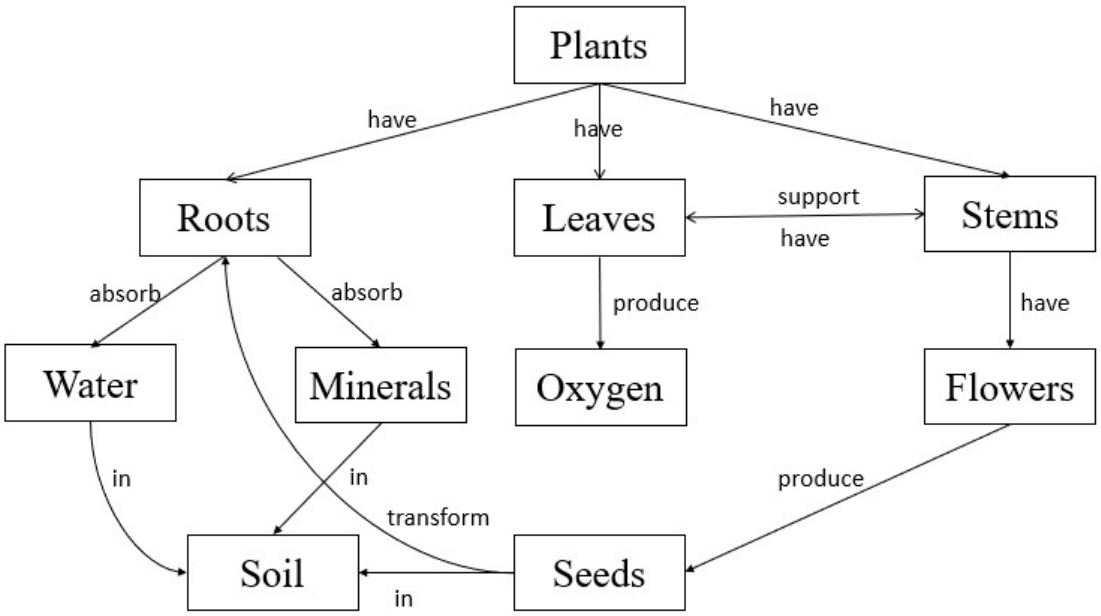


Figure 1: Example of semantic knowledge map reprinted from Kim et al. (2002)

Figure 1 is a semantic net illustrating plant compositions and relationships with the surrounding environment. It represents items and the relations among them, helping visualize the knowledge and providing a framework for knowledge management. The data on different plants, plant compositions, and living environments can be placed under the matching title. The meaning of each element in the data means is clear, providing better semantic clarity and interoperability among the data.

Natural language is our most prevalent knowledge representation system (Leinhardt 1987). It has been found that 90% of the world's data is held in unstructured formats that either do not have a pre-defined data model or that are not organized in a pre-defined manner (Alouini and Goldsmith 1999; Khan et al. 2010). Therefore, it is important to create a knowledge map that can provide semantic clarity, especially for information-intensive fields such as FM.

2. PROBLEM STATEMENT

As an information-intensive industry, FM is characterized by heterogenetic data (Mignard and Nicolle 2014). Efforts have been made to develop industrial standards for FM, such as IFCs developed by the Industry Alliance for Interoperability and COBie developed by the National Institute of Building Science (Parsanezhad and Dimyadi 2013). However, these standards lack semantic clarity, thereby hindering computational usage such as merging and interpreting facility data and homogenizing the representation of exchanges with building knowledge during a facility's life cycle (Chen and Luo 2016).

Moreover, the uptake for interoperability in the industry is slow with the cause generally attributed to human issues rather than technological issues (Aranda-Mena and Wakefield 2006; Karshenas and Niknam 2013). It has been found that despite the availability of digital information, people generally prefer to obtain information from colleagues with direct project knowledge, or from paper documents, and approximately 90% of the world's data is held in unstructured formats (Alouini and Goldsmith 1999; Khan et al. 2010).

Therefore, it is important to develop a knowledge map that serves as a visual aid that to show where knowledge can be found within a group or organization, and how to find those with the most expertise (Mindmanager 2018).

3. RESEARCH OBJECTIVES

The fundamental goal of this work was to create a knowledge map from unstructured text documents. Based on this goal and the methodology, the following objectives were developed:

1. Develop a process to extract knowledge items from text documents;

2. Develop a process to establish relations among the identified knowledge items;

3. Create a knowledge map based on the items and their relations and evaluate/validate the created knowledge map.

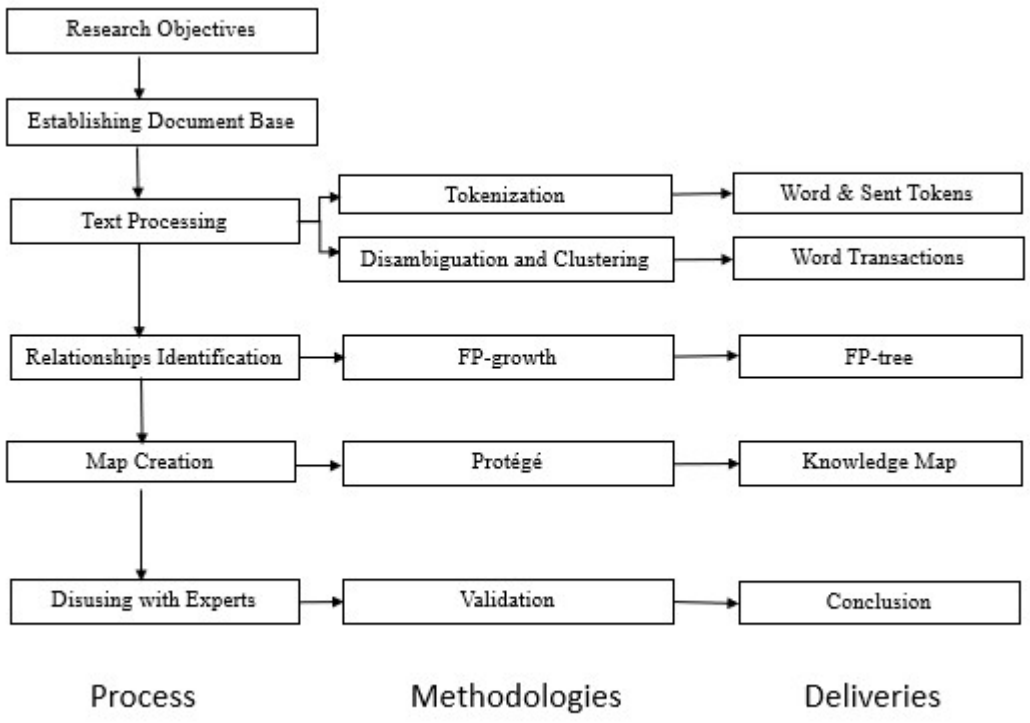


Figure 2: Research method

Figure 2 summaries the research processes. Based on the research objectives, the first step of this research was to collect the FM documents for natural language processing (NLP). The main processing methods included tokenization, disambiguation, and clustering. The clustered items were then mined by FP-Growth. Results were further organized and analyzed to create a semantic net in protégé to represent the knowledge in those documents. Finally, the map was reviewed by an expert.

4. LITERATURE REVIEW

4.1 Building Information Modeling for Facilities Management

FM comprises the operation and maintenance activities that ensure that end users receive the services for which the facility has been designed (Kuda and Berankova 2014). During a 40-year period, FM cost can be around 71% of the whole-life cost of a facility (Mckinstry 2013). An efficient FM project requires the cooperation of various actors from different domains, such as designers, constructors, and operators, and requires understanding and engaging different stakeholders, including end users and owners (Niskanen et al. 2014). The International Facility Management Association (IFMA) has defined 11 core competencies for FM: Communication, Emergency Preparedness and Business Continuity, Environmental Stewardship and Sustainability, Finance and Business, Human Factors, Leadership and Strategy, Operations and Maintenance, Project Management, Quality, Real Estate and Property Management and Technology (IFMA 2009). To meet different tasks and improve FM efficiency, an increasing number of digital technologies and tools such as BIM, BASs, and EMSs are involved. Due to the numerous heterogenous systems involved, FM-related data formats can vary greatly, which further increases the difficulty of information sharing and integration among the involved systems (Mignard and Nicolle 2014). Thus, it is a challenge for facility managers to effectively merge and interpret heterogeneous facility data and homogenize the representation of exchanges with building knowledge during a facility's life-cycle (Pittet et al. 2014).

The Architecture, Engineering, and Construction (AEC) industry is an information-intensive industry, and all related processes employed during different phases of a project including

planning, designing, building, manufacturing, occupying, and maintaining involve vast amounts of data used for a wide variety of purposes (Golabchi and Kamat 2013). Accessibility to required information is essential to any efficient FM and operations practice (Parsanezhad and Dimyadi 2013). BIM has demonstrated potential for tackling problems induced by insufficient access to information in all phases of a building's life-cycle including FM (Sabol 2008). BIM is a powerful shared-knowledge resource that stores data to support decision making about a facility through all the different phases in its life-cycle (Golabchi and Kamat 2013). After years of research and development, the AEC industry has now started to embrace and adopt BIM to support and promote the concepts of integration and interoperability (Halfawy and Froese 2007). However, BIM is mostly used in the design and construction phases, and for any implementation unrelated to the original intent of the BIM (design), organizations need to be able to represent data in a common interpretable form that provides the possibility of an accurate exchange of data among different software products and platforms, known as interoperability (Golabchi and Kamat 2013). Due to the unique nature of the AEC/FM industry, the development and deployment of systems integration and collaboration technologies are behind other sectors (Shen et al. 2010). The United States National Institute of Standards and Technology (NIST) estimated the cost of inadequate interoperability in the AEC/FM industry at $15.8 billion per year. (GCR 2004). The breakdown of this number reveals that the highest costs are incurred at the operations and maintenance phase.

4.2 Data Structure

As the reservoir, BIM contains all the information from design and construction that can be further used for FM. A BIM-enabled information system aims to seamlessly convey information from design and construction models to the FM sector. Technical solutions for optimizing information integration between BIM and FM software include using spreadsheets as document index tools (such as COBie), using IFCs format for exchanging information among BIM and FM systems, and using Application Programming Interfaces (APIs) and proprietary middleware such as EcoDomus, Onuma, Systems, and FM:Interact (Parsanezhad and Dimyadi 2013).

4.2.1 Industry Foundation Classes

Currently, the standard for representing, accessing, and sharing BIM information is IFCs (Karshenas and Niknam 2013). IFCs are developed by the Industry Alliance for Interoperability to enable interoperability among industry processes of all different professional domains in AEC/FM projects (Karola et al. 2002). In essence, IFC is a schema of EXPRESS, a standard modeling language used for product data (Chen and Luo 2016). IFCs include object specifications, or classes, and provide a useful structure for data sharing among applications. For instance, a door in IFC is not just a simple collection of lines and geometries; it has a door's attributes linked to a geometrical definition (Vanlande et al. 2008). However, IFCs only support a generalized BIM data structure in which data usage is not supported from the standpoint of interested parties in the project (Kang and Hong 2015). The main reason for this is that it lacks semantic clarity in mapping entities and relationships, which greatly hinders its application (Chen and Luo 2016).

4.2.2 Construction Operation Building Information Exchange

FM can provide a wide range of services for a building, including property management, financial management, management of human resources, health and associated risks, maintenance, and supplies management (Pittet et al. 2014). Due to the various scopes and functions in the FM phase, a facilities management system (FMS) can be used without being managed by a public data model (Kim et al. 2018). To define the specifications of data exchange for an FMS at the start of the operations & maintenance phase, the COBie system was developed by the National Institute of Building Science as a standard data exchange system throughout a building's lifecycle to ensure the interoperability of BIM data for an FMS. It aims to facilitate the transfer of digital information from the design and construction process to the FM databases (East 2007). It provides guidelines of consistent procedures for file naming, data storage, data indexing, and data archiving so that information can be easily retrieved and validated (Smith and Tardif 2009). For each building project, the COBie file contains the information of all phases of the project. The owner or facility manager can load information about operations and maintenance directly into a spreadsheet software or a COBie-based maintenance management system, either at the beginning of the building operation or for updating data.

BIM standards such as IFC and COBie have been recognized by construction, and political actors as a highly promising tool for resolving issues of data dematerialization and for enhancing building information interoperability in the AEC/FM field (Farias et al. 2015). However, lack of semantic information greatly hinders their application (Chen and Luo 2016). To facilitate intelligent uses such as information integration and ontology reasoning, semantic clarity is needed in mapping entities and relationships.

4.3 Ontology and Semantic Web

In recent years, more and more efforts have been made to research ontology-driven information systems. Compared to conventional information systems that are built on top of a relational database, ontology-driven information systems are built on ontology-based content models and adopt a highly interdisciplinary approach (Guarino 1998).

Ontology is an explicit, partial account of a conceptualization and the intended models of a logical language (Giaretta and Guarino 1995). Borst (1997) further defined it as "a formal specification of a shared conceptualization". In other words, it provides the kinds and structures of objects, properties, events, processes and relations within the defined domain (Smith and Welty 2001). It is "a collection of vocabularies and the specifications of the conceptualization of a given domain" (Tari 2008). An ontology defines the basic terms and relations comprising the vocabulary of a topic area as well as the rules for combining terms and relations among terms (Sugumaran and Storey 2002).

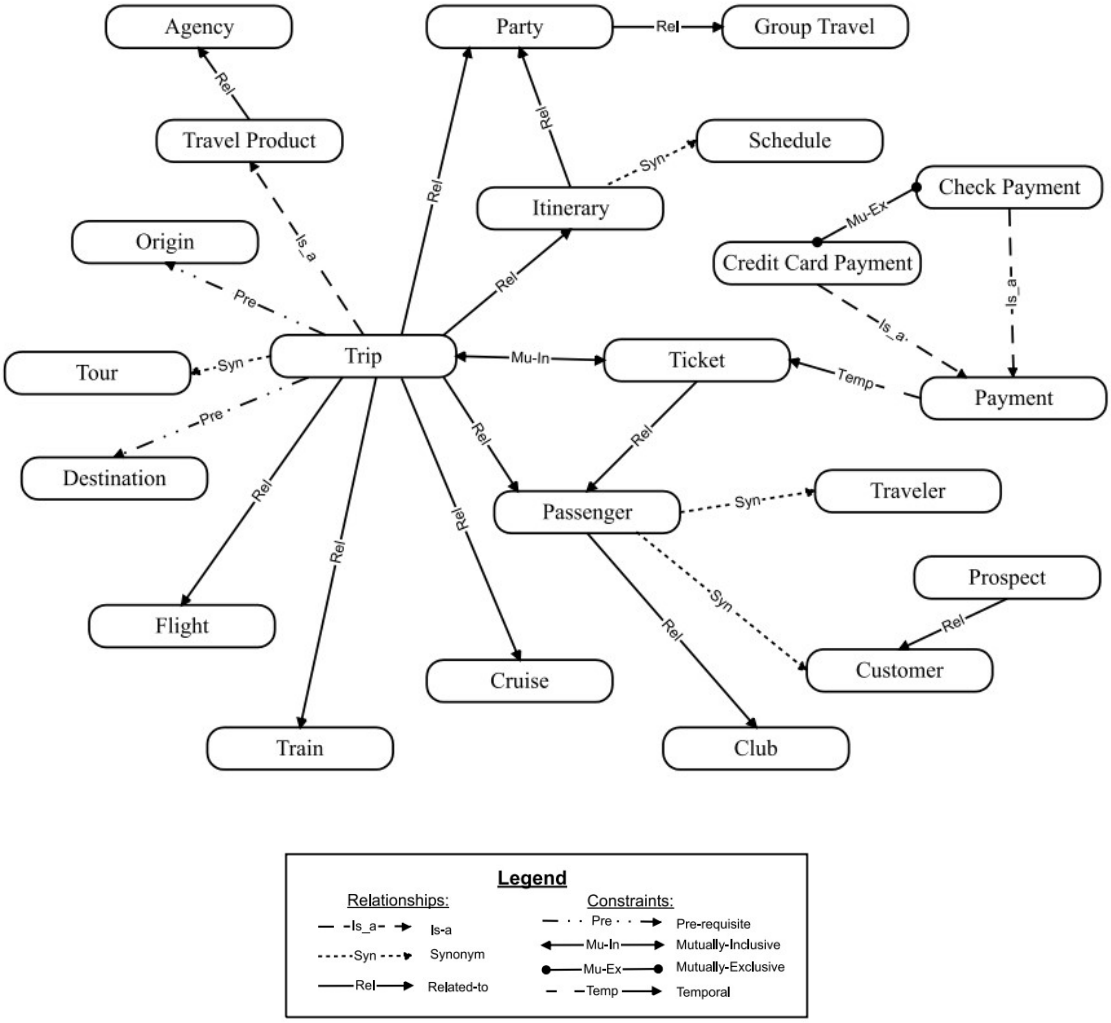


Figure 3: Ontology example reprinted from Sugumaran and Storey (2002)

Figure 3 shows an example of a simple ontology for the online travel domain. It includes concepts and relationships and constraints among those concepts. For instance, the trip is a travel product related to train, cruise, and passenger.

The goal of a domain ontology is to capture domain knowledge and create semantics explicitly in a generic way, providing the basis for agreement within the domain (Baader et al. 2005). It is expressed in a knowledge representation programming languages such as Resource

Description Framework (RDF), and Web Ontology Language (OWL), which provide a formal frame of semantics and enable interoperation among applications. Domain ontology can be further used for data sharing, knowledge reusing, and information reasoning and querying. By applying the Semantic Web and ontology technologies, it is possible to define a comprehensive facility data model as a metadata layer that classifies and describes relevant data within the domain of interest. This brings in advantages not only in improving interoperability and reducing system heterogeneity, but also in better downward and upward compatibility of technical systems and accompanying software (Tomašević et al. 2015).

Although academic efforts have been made for improvement, the uptake for interoperability in the industry is still slow, generally attributed to human issues rather than technological issues (Aranda-Mena and Wakefield 2006; Karshenas and Niknam 2013). For example, AEC/FM engineer teams do not exploit IFCs, and there is no central IFC database or tools for IFC analysis, comparison, or visualization during construction and FM activities (Vanlande et al. 2008). It also has been found that despite the availability of digital information, people generally prefer to obtain information from colleagues with direct project knowledge, or from paper documents and approximately 90% of the world's data is held in unstructured formats (Alouini and Goldsmith 1999; Khan et al. 2010). As more digital information is amassed, including information from COBie and BIM models, the need is fully apparent for automatic retrieval of useful knowledge from the huge amount of textual data, especially for FM of information-intensive business processes (Anderson et al. 2012; Fong et al. 2004; Toledo-Alvarado et al. 2012).

4.4 Ontology Creation Methods

Most ontology and database creation is carried out on a manual basis, with desgin quality highly dependent on the skill and expertise level of the designer (Sugumaran and Storey 2002). There are many guidelines and anecdotal experiences reported in the literature; however, there is no proposal for a general methodology for building ontologies (Uschold and King 1995). The main methodologies developed include IDEF5, KACTUS, METHODOLOGY, On-To-Knowledge and so on (Corcho et al. 2003). Some significant steps in those methodologies are capture, coding, and evaluation.

In the capture stage, the key concepts and relationships in the domain of interest are identified, and precise unambiguous text definitions are produced for such concepts and relationships. In the coding stage, the conceptualization captured in the capture stage is explicit represented in some formal language after choosing a representation language and creating the codes.

A knowledge representation language provides a formal frame of semantics and enables interoperation among applications. RDF Schema(RDFS), OIL, DAML+OIL, and OWL are modeling web languages that have been developed to represent or express ontologies. Though these description languages are able to form complex class expressions and axioms (Baader et al. 2005), they have different terminologies and expressions and some can represent certain logical relations while others cannot. Among them, OWL is particularly significant because it was developed by the World Wide Web Consortium (W3C) Web Ontology Working Group and is an official recommendation of W3C (Horrocks and Patel-Schneider 2004). In general, though, most of them are based on XML syntax, OWL is a language that builds on RDF and RDFS using developed XML syntax developed. It uses the RDF meaning of classes and

properties. (Taye 2011). Protégé is a free, open-source platform developed by Standford University to provide users with a suite of tools to construct domain model and knowledge-based applications with ontology. It provides a feature-rich ontology editing environment with full support for OWL (University 2016). Protégé and OWL were the main tool and language, respectively, for building the knowledge map, and the created knowledge map was evaluated by the Property Management unit of Financial Management Operations, which wrote and maintains the documents.

4.5 Fundamental Concepts of Natural Language Processing

A natural language is any language that has evolved naturally in humans through use and repetition without conscious planning or premeditation (Langendoen 1993). While natural language is our most prevalent knowledge representation system, there are others including pictures, diagrams, graphs and maps, which may be more "computationally efficient" than natural language—for example, facilitating faster search and recognition of relevant information (Leinhardt 1987). A knowledge map is one of those alternative knowledge representation systems and has been applied in various disciplines. For example, Novak and Gowin created a system in education; Alexodro proposed cognitive maps in management and Toulmin developed a theory of science argument based on typed concept maps (Gaines and Shaw 1995). The need for knowledge management has been identified both in FM practice and FM research (Nenonen et al. 2014).

4.5.1 Natural Language Processing

Natural language processing (NLP) is a computerized approach to analyzing text based on both a set of theories and a set of technologies (Liddy 2001). It aims to develop appropriate tools and technologies to make computer systems understand and manipulate natural languages (Chowdhury 2003). The difficulty of processing natural language is the pervasive ambiguity including lexical ambiguity ("duck" can be a noun and a verb), structural or syntactic ambiguity, semantic ambiguity, pragmatic ambiguity, and referential ambiguity (Allen 2003). Basic NLP tasks include tokenization and parsing, lemmatization/stemming, part-of-speech tagging, language detection, and identification of semantic relationships (SAS 2017).

4.5.2 Tokenization

Tokenization is the beginning step for natural language processing NLP (Webster and Kit 1992). Tokenization is the task of chopping up the text content into pieces called tokens. Tokens often are referred to loosely as terms or words (Manning et al. 2008). Tokens are the basic units, on which further analysis and generation can be carried.

4.5.3 Tagging and Annotation

Tagging and annotation is the second step in the typical NLP pipeline, following tokenization (Bird et al. 2009). It labels the tokens based on the different principles and metadata tags. For example, the sentence Messi scored 3 goals for FC Barcelona in Camp Nou can be tagged as Messi|FW, Score|VBD, 3|CD, goals|NNS, for|IN, FC|NNP, Barcelona|NNP, in|IN, Camp|NNP, Nou|NNP in part-of-speech tagging; or as Messi|Person, 3|Number, FC Barcelona|Organization, Camp Nou|Location in named entity reorganization annotation. Having a good annotation scheme and accurate annotations is critical for machine learning that relies on data outside of the text itself (Pustejovsky and Stubbs 2012).

4.5.4 Parsing

In the parsing task, a parse tree is constructed to filter out untargeted information. Some parsers use the existence of a set of grammar rules to parse, but recently more and more parsers deduce the parse trees directly from the given data using complex statistical models (Bikel and Marcus 2004). Most parsers also operate in a supervised setting and require the sentence to be tagged before it can be parsed (Madnani 2007).

4.5.5 Frequent Itemset Mining

Frequent itemset mining (FIM) includes Apriori and Frequent Pattern (FP)-Growth and is a useful tool for discovering frequently co-occurrent items (Li et al. 2008). In FIM, the base data take the form of sets of instances (also called transactions), each with a number of features (also called items) (Drachen 2012).

Table 1: Frequent item mining example reprinted from Sagonas (2006)

Transaction	Items
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Bread, Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Table 1 is an example of an itemset. It contains five transactions and each transaction has different items in it. The task for the FIM algorithm is then to find all common sets of items and

connect them based on the transactions. A minimum frequent threshold is defined. Any item that recurs less than the threshold is considered nonfrequent and is excluded from the mining.

5. RESEARCH METHODOLOGY

5.1 Data Collection

The first step was establishing the text corpus consisting of FM documents including the procedures manual, asset management manual, and inventory manual. The procedures manual is a document written to support a "policy directive," describing who, what, where, when, and why corporate accountability is established in support of the implementing a "policy." The purpose of this manual is to set forth the overall procedures for property management within the organization. This document consists of nine chapters listing the liabilities of individuals and departments, as well as the processes for inventory management and property management. Combined with the asset management manual and inventory manual, the procedures manual describes the overall process for the property management process within the organization.

5.2 Data Analysis

5.2.1 Text Processing

Based on the research methodology, the first step to develop semantic knowledge maps from text data was to extract keywords from the data. NLP strategies such as tokenization, stop words, and stemming were used for extraction. The second task was to disambiguate and cluster the keywords by the keyword-in-context method using WordNet. This step cleaned the keywords by combining keywords with similar meanings, increasing the accuracy of the final model.

Natural Language Toolkits (NLTK) developed by the University of Pennsylvania is an open-source library for NLP (Bird and Loper 2004). NLTK consists of interdependent modules designed for different NLP purposes, including tokenization modules, parsing modules, tagging modules, text classification modules, visualization modules, etc. (Loper and Bird 2002). Python is widely chosen as the implementation language for NLTK because it has a shallow learning curve, transparent syntax and semantics, and good string-handling functionality (Bird et al. 2008). Python is a simple yet powerful (and free) programming language with excellent functionality for processing linguistic data (Bird et al. 2009). Python combined with NLTK is a powerful tool to learn and conduct research in NLP (Madnani 2007).

5.2.2 Finding Relations in Contexts

FP-Growth is a rule-based learning algorithm proposed by Han et al. (2000) and is used to map the relations among different contexts. The FP-Growth algorithm is currently one of the currently fastest and most popular FIM algorithms (Borgelt 2005; Li et al. 2008). In testing and some later works, this method proved to outperform other popular methods for mining frequent patterns. FP-Growth uses a divide-and-conquer strategy. It first compresses the input database, creating an FP-Tree instance to represent frequent items, and then divides the compressed database into a set of conditional databases, each one associated with one FP; finally, each such database is mined separately (Han et al. 2011; Han et al. 2000). FP-Growth involves two scans. The first scan determines the frequencies of the items and deletes those appearing in fewer transactions than a user-defined threshold. Then FP-Growth starts to mine the FP-Tree of the projected database (Borgelt 2005). Table 2 and Figures 4 illustrate how FP-Growth works.

Table 2: Example of FP-Growth table

Transaction	Items
1	A,B,D,E
2	B,C,E
3	A,B,D,E
4	A,B,C,E
5	A,B,C,D,E
6	B,C,D

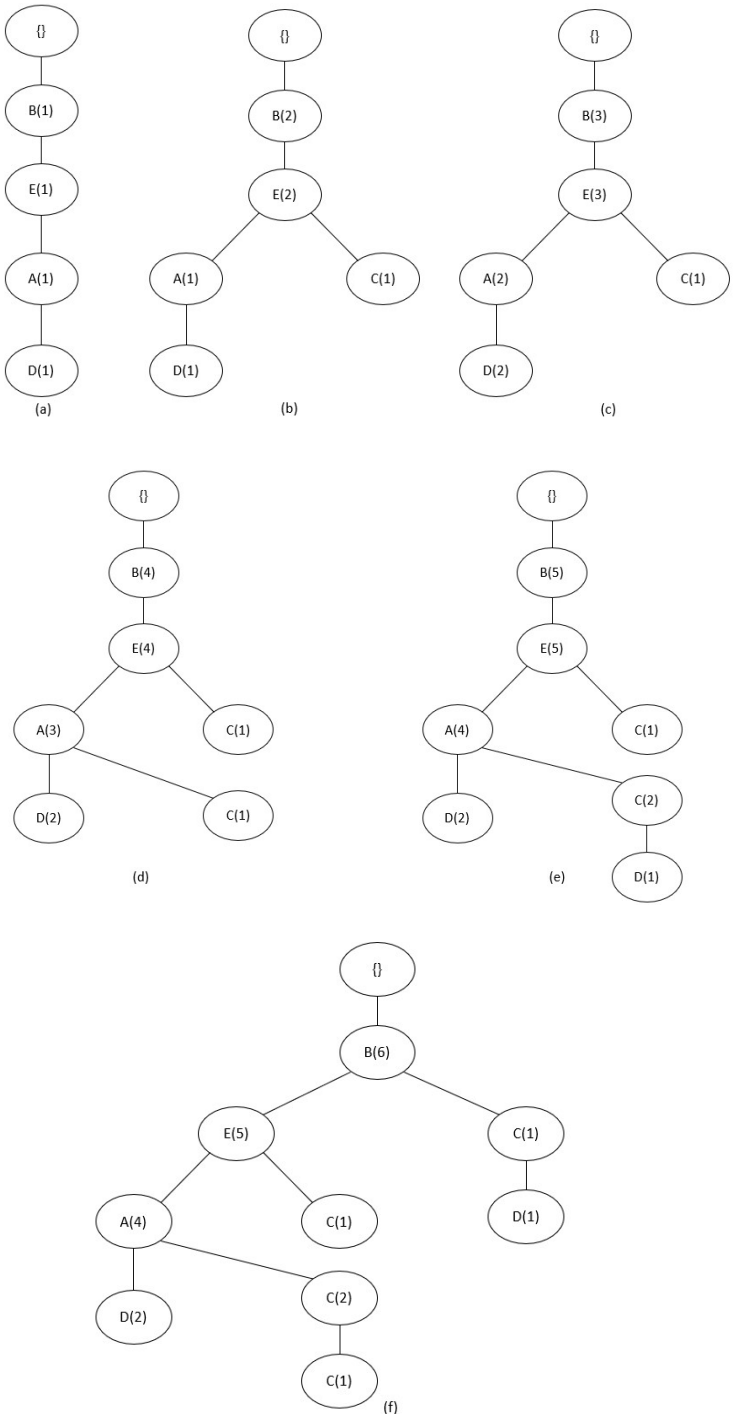


Figure 4: Process of FP-tree development

Table 2 contains five transactions: A,B,D,E; B,C,E; A,B,D,E; A,B,C,E; A,B,C,D,E; and B,C,D. The algorithm first scans the table and creates a list of frequencies {B(6), E(5), A(4), C(4), D(4)}. Thereafter, the FP-Tree is iteratively constructed for each transaction using the sorted list of items (see Figures 3 for illustration).

5.2.3 Trimming and Verification

The last step of the knowledge map creation process was to use Protégé to build the OWL model. The identified keywords and their relations were imported into Protégé to build the OWL model. Results illustrate the relations among the identified keywords in the unstructured contents and visualize them. The final map was shown to an expert to verify the accuracy and usefulness of the knowledge map.

6. ANALYSIS AND FINDINGS

6.1 Text Mining

The three documents used in this research were kept as PDF files. Though PDF is a convenient, secure, and graphic-integrity format containing the same text, in order to manipulate the content more easily, the first step was to chop the text content into smaller pieces.

```
with open('procedures_manual.pdf','rb') as pdf_file, open('a.txt', 'w') as text_file:
    read_pdf = PyPDF2.PdfFileReader(pdf_file)
    number_of_pages = read_pdf.getNumPages()
    for page_number in range(7,133):   # use xrange in Py2
        page = read_pdf.getPage(page_number)
        page_content = page.extractText()
        text_file.write(page_content)
```

Figure 5: Saving content in a txt file

Figure 5 shows the code for extracting the content from those documents and saving them in a txt file. The code allows extracting the content from a range and lets the model read the PDF files, extract text contents, and save the contents into a .txt file. It allows controlling the range of contents by typing in the targeted page numbers. This process included using PyPDF, a pure-Python PDF library capable of splitting, merging, cropping, and transforming the pages of PDF files (mstamy2 2011). It was first launched by Mathieu Fenniak in 2005, and in 2011, Phaseit, after consulting with Mathieu and others, sponsored PyPDF2 as a fork of pyPdf on GitHub.

After loading the document into Python, the next step was to continue to chop the content into words and sentences called tokens; this step is called tokenization.

```
tokens = word_tokenize(file_content)
sentences = sent_tokenize(file_content)
```

Figure 6: Word and sentence tokenization

Figure 6 shows two functions of tokenization: word_tokenization and sent_tokenization. Word_tokenization is a wrapper function in nltk.tokenize module that tokenizes text into words by the TreebankWordTokenizer (TextMiner 2014). Sent_tokenization uses an instance of PunktSentenceTokenizer from the nltk.tokenize.punkt module. This instance has already been trained on and works well for many European languages (TextMiner 2014), which means that the function knows what punctuation and characters mark the end of a sentence and the beginning of a new sentence. It can split a document into sentences.

Stop words are words filtered out from the text data. Though “stop words” usually refers to the most common words in a language, there is no single universal list of stop words used by all NLP tools, and indeed, not all tools even use such a list (Mishra and Vishwakarma 2017). Any group of words can be chosen as the stop words for a given purpose.

```
stop_words = set(stopwords.words("english"))
```

```
print (stop_words)
```

```
{'all', 'to', 'll', 'that', 'such', 'if', 'itself', 'other', 'him', 'don', 'in', 'own', "you've", 'doing', 'y', 'm',
've', 'has', 'which', 'she', 'the', 'as', 'no', 'those', 'why', "should've", 'some', 'down', "that'll", 'here', 'betw
een', 'weren', 'wasn', 'an', 'will', 'we', 'do', "wouldn't", 'too', 'yours', 'after', 'himself', 'these', 'both', 'he
r', 'than', 'again', 'was', 'doesn', "isn't", 'whom', 'and', 'then', 'themselves', "shan't", 'nor', 'needn', 'mustn',
'out', 'into', 'off', 'hadn', 'hasn', 'he', "couldn't", 'for', 'hers', 'but', 'more', 't', "mustn't", "haven't", "do
n't", 'isn', 'at', "you'll", 'now', 'our', 'did', 'this', "she's", 'me', 'o', 'about', 'couldn', 'theirs', 'until',
'them', 'only', "didn't", 'through', 'been', 'there', "doesn't", 'herself', 'by', 'haven', 'were', 'each', 'mightn',
'during', 'am', 'from', 'when', 're', 'any', 'just', 'up', 'once', "hasn't", 'how', 'further', 'very', 'because', "we
ren't", 'shouldn', 'their', 'i', 'won', 'does', 'a', 'myself', 'd', 'same', 'or', 'having', "mightn't", 'who', "was
n't", 'before', 'yourself', 'yourselves', 'with', 'wouldn', 'its', 'can', 'of', 'his', 'against', 'few', 'over', 'whe
re', 'ain', 'ma', 'your', 'what', 'is', 'below', 'ours', "aren't", 'didn', "you're", 'shan', 'above', 'are', 'aren',
"won't", 'be', 'had', 'my', 'they', 'on', "shouldn't", 'being', "needn't", 'most', 'should', 'you', 'have', "it's",
'it', 'while', 'not', 'so', 'under', 's', "hadn't", 'ourselves', "you'd"}
```

```
stop_words.add('The')
stop_words.add(',')
stop_words.add('-')
stop_words.add('.')
```

Figure 7: Stop words

Though NLTK provides a bag of words usually used as stop words, after analyzing the documents, some symbols and words were added. Figure 7 shows the default stop words in the NLTK as well as how to add new stop words to filter out non-useful tokens.

Collocations in NLP are expressions of multiple words that commonly co-occur. The collocations package in NLTK provides collocation finders that, by default, consider all ngrams, a contiguous sequence of n items from a given sample of text or speech in a text (Broder et al. 1997), as candidate collocations. Collocations help to find ngrams that commonly occur together, meaning they are likely to be one item and should be analyzed together.

```
from nltk import collocations
bigram_measures = collocations.BigramAssocMeasures()
bigram_finder = collocations.BigramCollocationFinder.from_words(tokens)

# Filter to top 20 results; otherwise this will take a LONG time to analyze
bigram_finder.apply_freq_filter(20)
for bigram in bigram_finder.score_ngrams(bigram_measures.raw_freq)[:10]:
 print (bigram)
```

```
(('A', '&'), 0.009429434516247424)
(('&', 'M'), 0.00934473899664041)
(('Texas', 'A'), 0.009231811637164394)
(('of', 'the'), 0.005872222692752887)
(('Property', 'Management'), 0.0057310634934078656)
(('to', 'the'), 0.00423477598035064)
(('.', 'The'), 0.003895993901922588)
(('FDP', '-'), 0.0038395302221845796)
(('M', 'Property'), 0.0035854436633635414)
(('February', '2016'), 0.003557211823494537)
```

(a) Collocation before combing ngrams

```
token = re.findall(r'A&M|Texas A&M|Property Management|Procedures Manual|Procedures Manager\S+',file_content)
```

(b) Combine collocated ngrams

```
bigram_finder = collocations.BigramCollocationFinder.from_words(filtered_words)

# Filter to top 20 results; otherwise this will take a LONG time to analyze
bigram_finder.apply_freq_filter(1)
for bigram in bigram_finder.score_ngrams(bigram_measures.raw_freq)[:10]:
 print (bigram)
```

```
(('Property Management', 'Texas A&M'), 0.2647058823529412)
(('Texas A&M', 'Property Management'), 0.2647058823529412)
(('Texas A&M', 'Texas A&M'), 0.14705882352941177)
(('Property Management', 'Property Management'), 0.11764705882352941)
(('A&M', 'Property Management'), 0.058823529411764705)
(('Property Management', 'A&M'), 0.058823529411764705)
(('A&M', 'Texas A&M'), 0.029411764705882353)
(('Texas A&M', 'A&M'), 0.029411764705882353)
```

(c) Collocation after combing ngrams

Figure 8: Collocation analysis

Figure 10 shows the collocation analysis. As Figure 8(a) shows, terms such as Texas A&M, property management, and procedures manual were separated in the tokenization process. The Re.findall function was used to combine the separated words together. After the combination in Figure 8(b), the new result in Figure 8(c) shows that obvious combination anymore.

The last step in the text mining stage was to group keywords. This was based on the theory that if the words exist in one sentence, there is some kind of existing relation.

```
sentences = "My name is sing song. I am a mother. I am happy. You sing like my mother".split(".")
search_keywords=['mother','sing','song']

for sentence in sentences:
    lst = []
    for word in search_keywords:
        if word in sentence:
            lst.append(word)
    print('{0} key word(s) in sentence: {1}'.format(len(lst), ', '.join(lst)))
    print(sentence + "\n")
```

```
2 key word(s) in sentence: sing, song
My name is sing song

1 key word(s) in sentence: mother
 I am a mother

0 key word(s) in sentence:
 I am happy

2 key word(s) in sentence: mother, sing
 You sing like my mother
```

Figure 9: Word pairing

Figure 9 shows the code and an example of word pairing. The codes look for the predefined keywords by sentences and paired them. The paired words were further used as transactions for FP-Tree creation.

6.2 Creating the FP-Tree

The words were grouped in each sentence in the above step. The grouped words constitute transactions. The next step was to use FP-Growth to go through the transactions and create relations among the transactions.

```
class treeNode:
    def __init__(self, nameValue, numOccur, parentNode):
        self.name = nameValue
        self.count = numOccur
        self.nodeLink = None
        self.parent = parentNode
        self.children = {}
    def inc(self, numOccur):
        self.count += numOccur
    def disp(self, ind=1):
        print ('  '*ind, self.name, ' ', self.count)
        for child in self.children.values():
            child.disp(ind+1)
```

Figure 10: Defining FP-Tree node

Figure 10 shows the code used to create the TreeNode. The function Self.nodeLink = None was used to connect similar items. Self.children = {} is the space used to store the nodes. The disp function displays the content in the tree form.

```
def createTree(dataSet, minSup=1): #create FP-tree from dataset but don't mine
    headerTable = {}
    #go over dataSet twice
    for trans in dataSet:#first pass counts frequency of occurance
        for item in trans:
            headerTable[item] = headerTable.get(item, 0) + dataSet[trans]
    for k in list(headerTable):  #remove items not meeting minSup
        if headerTable[k] < minSup:
            del(headerTable[k])
    freqItemSet = set(headerTable.keys())
    #print 'freqItemSet: ',freqItemSet
    if len(freqItemSet) == 0: return None, None  #if no items meet min support -->get out
    for k in headerTable:
        headerTable[k] = [headerTable[k], None] #reformat headerTable to use Node link
    #print 'headerTable: ',headerTable
    retTree = treeNode('Null Set', 1, None) #create tree
    for tranSet, count in dataSet.items():  #go through dataset 2nd time
        localD = {}
        for item in tranSet:  #put transaction items in order
            if item in freqItemSet:
                localD[item] = headerTable[item][0]
        if len(localD) > 0:
            orderedItems = [v[0] for v in sorted(localD.items(), key=lambda p: p[1], reverse=True)]
            updateTree(orderedItems, retTree, headerTable, count)#populate tree with ordered freq itemset
    return retTree, headerTable #return tree and header table
```

Figure 11: Creating FP-Tree

Figure 11 shows the algorithm for creating the FP-Tree. The algorithm goes through the dataset twice. The first pass counts the frequency of occurrences, removes items that do not meet the minimum threshold, and creates a header table. The second count puts transaction items in order and creates the FP-Tree.

```
def createInitSet(dataSet):
    retDict = {}
    for trans in dataSet:
        retDict[frozenset(trans)] = 1
    return retDict
```

Figure 12: Updating FF-tree

The createTree() function doesn't take the input data as listed; it expects a dictionary, with the itemsets as the dictionary keys and the frequency as the value. The createInitSet() function does this conversion. The complete algorithm is shown in Appendix A.

6.3 Results Analysis

The Procedures Manual has nine chapters: Delegation of Responsibility; Property Reporting Requirements; Inventory Items and Coding; Tagging of Equipment; Property Acquisition; Property Dispositions; Property Transfers; Miscellaneous Property Situations; and Managing the Inventory. The asset management manual and inventory manual provide supplemental description of the procedures manual. For example, in the inventory manual, detailed information is provided about Chapter 3, Inventory Items and Coding. It supplies more detailed information about the rules to assign codes and the meanings of each code series. For instance, classroom is 100 series with 110A for general classroom and 110B for teaching auditorium.

The documents were separately analyzed and also were analyzed chapter by chapter and combined together to form the final knowledge map. This type of analysis reduced the influence between chapters because the topics of each chapter is different.

Table 3: Chapter 1 frequent words

Terms	Frequency
Property	81
Department	35
Responsibility	26
TAMU	23
Head	15
Employee	14
FDP	13
Asset	12
Attorney	8
Inventory	7
FAMIS	7
Manager	7

Table 3 shows the frequent words identified from Chapter 1. The words then were set as the keywords to create the transactions from the document. Transactions are words that appearing in the same sentence indicating potential relations between them.

```
[['TAMU', 'property'],
 ['head', 'property'],
 ['employee', 'property', 'manager'],
 ['TAMU', 'responsibility', 'property', 'asset'],
 ['property', 'manager'],
 ['TAMU', 'responsibility', 'property', 'asset'],
 ['TAMU', 'responsibility', 'head', 'asset'],
 ['TAMU', 'property', 'inventory'],
 ['property', 'manager'],
 ['employee', 'asset', 'inventory'],
 ['TAMU', 'responsibility', 'employee', 'asset', 'department'],
 ['TAMU', 'property', 'DPC'],
 ['head', 'FAMIS', 'property', 'DPC'],
 ['head', 'property', 'DPC'],
 ['TAMU', 'responsibility', 'property', 'FDP'],
 ['head', 'DPC'],
 ['TAMU', 'property', 'FDP', 'DPC', 'inventory'],
 ['responsibility', 'head', 'property', 'FDP'],
 ['responsibility', 'empolyee', 'property'],
 ['TAMU', 'property', 'head'],
 ['head', 'inventory'],
 ['property', 'asset', 'inventory'],
 ['property', 'DPC'],
 ['FAMIS', 'property', 'DPC', 'TDP', 'department'],
 ['FAMIS', 'DPC']]
```

Figure 13: Word transactions extracted from Chapter 1of the procedures manual

Figure 13 shows the transactions from Chapter 1. Then the transaction were put into the FP-Growth algorithm to create the FP-Tree.

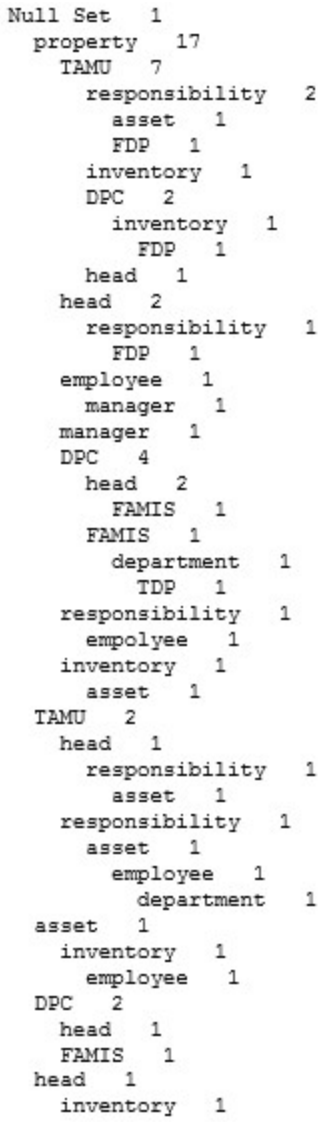


Figure 14: FP-Tree for Chapter 1 of the procedures manual

Figure 14 shows the FP-Tree created. The tree is displayed horizontally. Each indent indicates a branch.

```
property   17
  TAMU   7
    responsibility   2
      asset   1
      FDP   1
    inventory   1
    DPC   2
      inventory   1
        FDP   1
    head   1
```

Figure 15: Part of the FP-Tree for Chapter 1 of the procedures manual

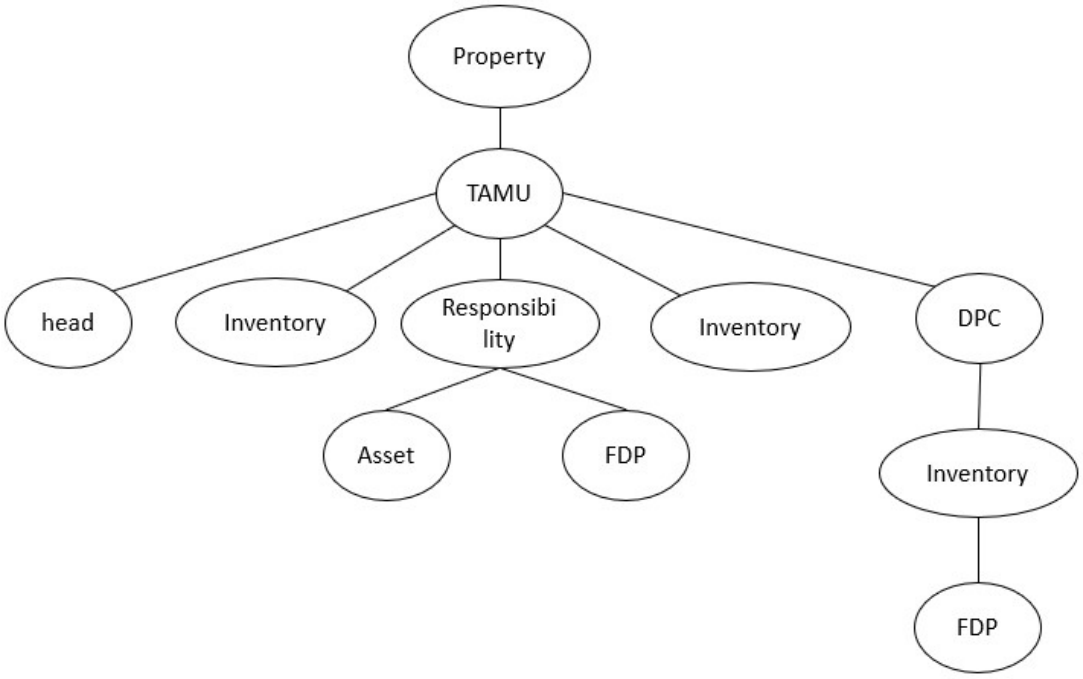


Figure 16: Indicated relations of the FP-Tree

Figure 15 shows part of the FP-Tree for Chapter 1 of the procedures manual. It includes six transactions: [Property, TAMU, head], [Property, TAMU, Inventory], [Property, TAMU, Responsibility, Asset], [Property, TAMU, Responsibility, FDP], [Property, TAMU, Inventory], and [Property, TAMU, DPC, Inventory, FDP] and Figure 16 illustrates the relations with a

semantic net. An item of note is that the map does not show a hierarchy relation. It only indicates that there is a certain relation between the connected nodes.

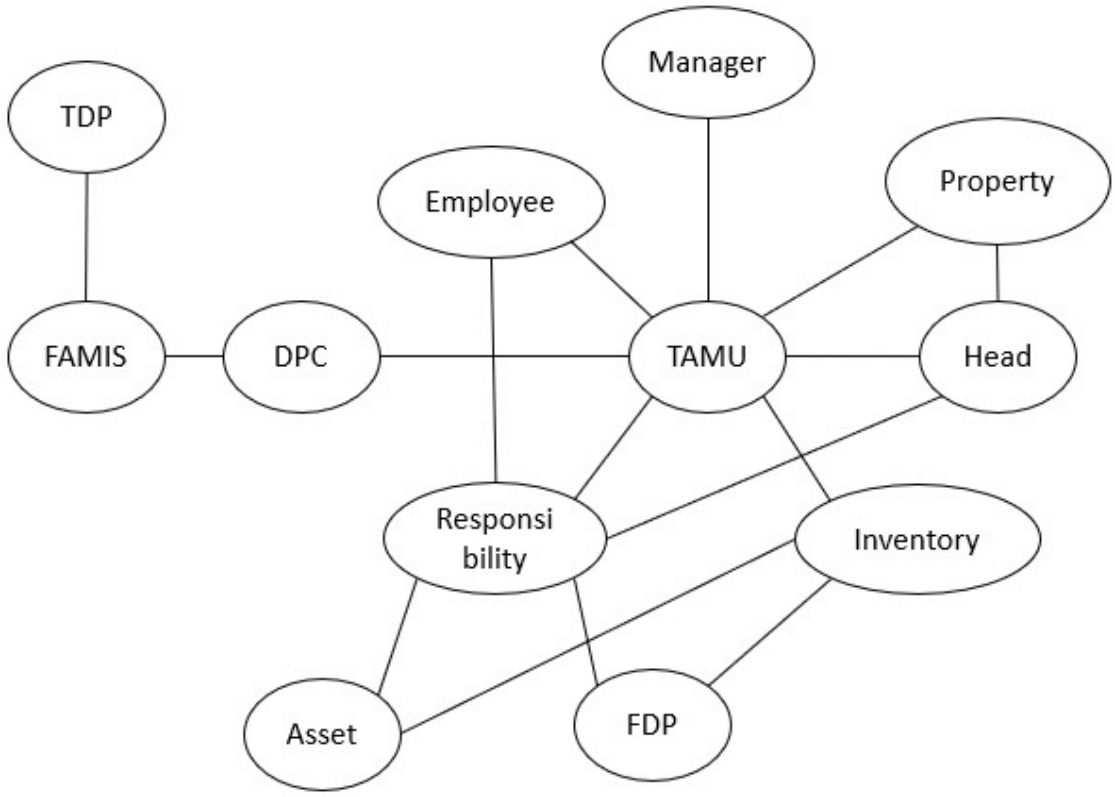


Figure 17: Chapter 1 knowledge map

The results from the FP-Tree were manually trimmed and reorganized. Figure 17 illustrates the whole items and relations identified from Chapter 1.

Table 4: Summary of keywords in the procedures manual

Chapters / Terms Identified from the chapter	Chapter 1	Chapter 2	Chapter 3	Chapter 4	Chapter 5	Chapter 6	Chapter 7	Chapter 8	Chapter 9	Total times of appearance
Property	81	38	23		84	81	47	41	45	440
Department	35	28			23	33	39	16	72	246
Responsibility	26									26
TAMU/A&M	23	25		16	73	40	40	3	25	245
Head	15									15
Employee	14							3		17
FDP	13	23			31	30	14	13		124
Asset	12		64	27	85	32	4	12	39	275
Attorney	8									8
Inventory	7	41	19	22	18	18	11	11	93	240
FAMIS	7	4	29	6	17			7	24	94
Manager	7									7
Scan		17								17
Codes		4	51		25		5	3	10	98
Item			23	4		11	7	4	20	69
Canopy			10				3	5	21	39
DPC			11						59	70
Bar code				8						8
Tag				27	12					39
number				32						32
equipment				17	28			18	49	112
Acquisition					16		6			22
purchase					37					37
stolen						16				16
missing						19				19
Surplus							28			28
System							15			15
Transfer							20			20
411A							10			10
warehouse							8			8
software								34		34
location								10	7	17

Table 4 shows all the keywords identified from the Property Management unit's procedures manual. New keywords were identified from each chapter as the chapter topic changed. With new items identified, the knowledge map grew.

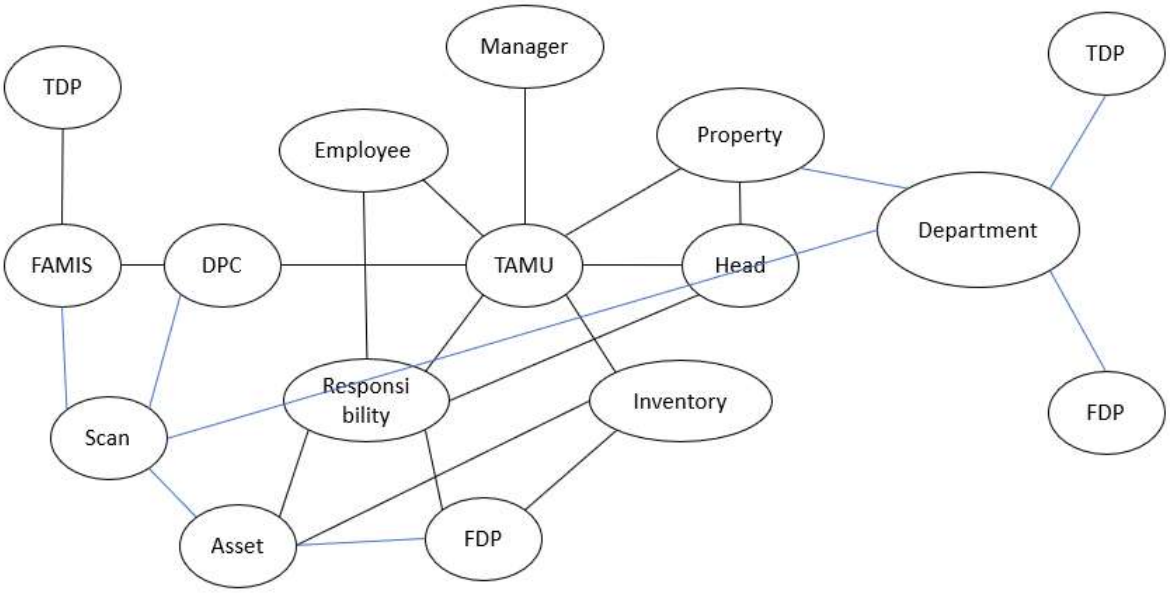


Figure 18: The knowledge map after including the transactions from chapter 2

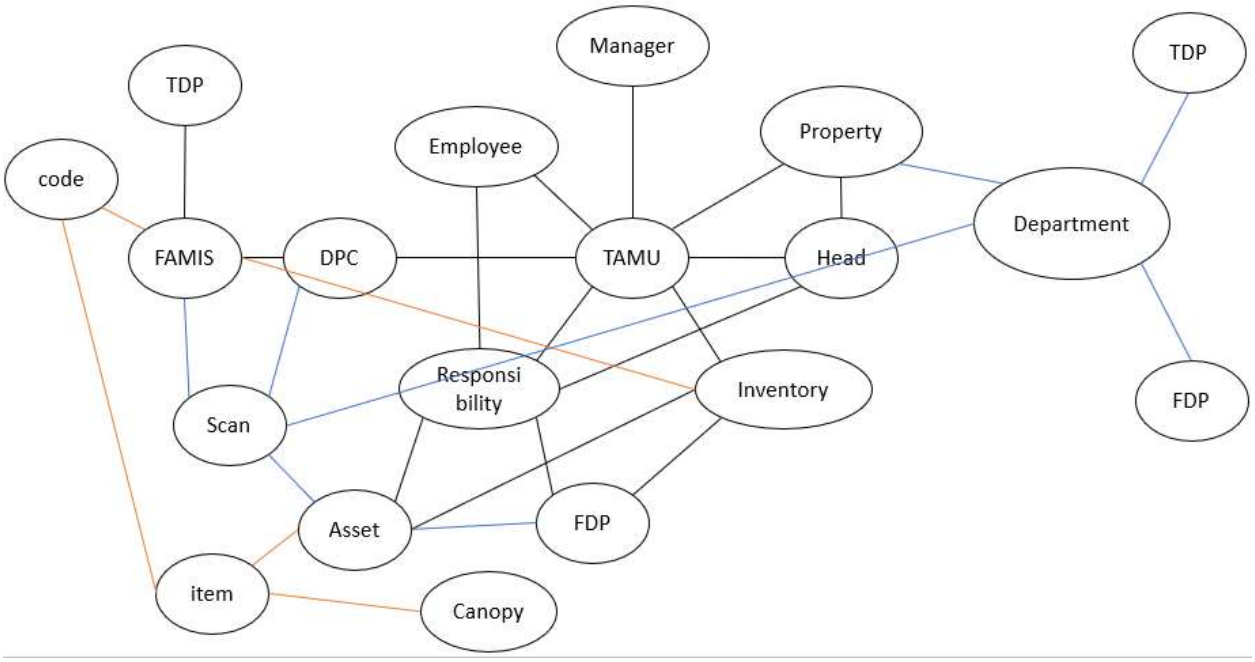


Figure 19: The knowledge map after including the transactions from chapter 3

Figures 18 and 19 show how the net grew with more chapters analyzed.

After finishing mapping the procedures manual, the asset manual, and inventory manual were analyzed based on the same principle.

The last step was to create the knowledge map in Protégé. Protégé is an open-source ontology editor. It provides frameworks and tools for data management and different query and semantic web languages.

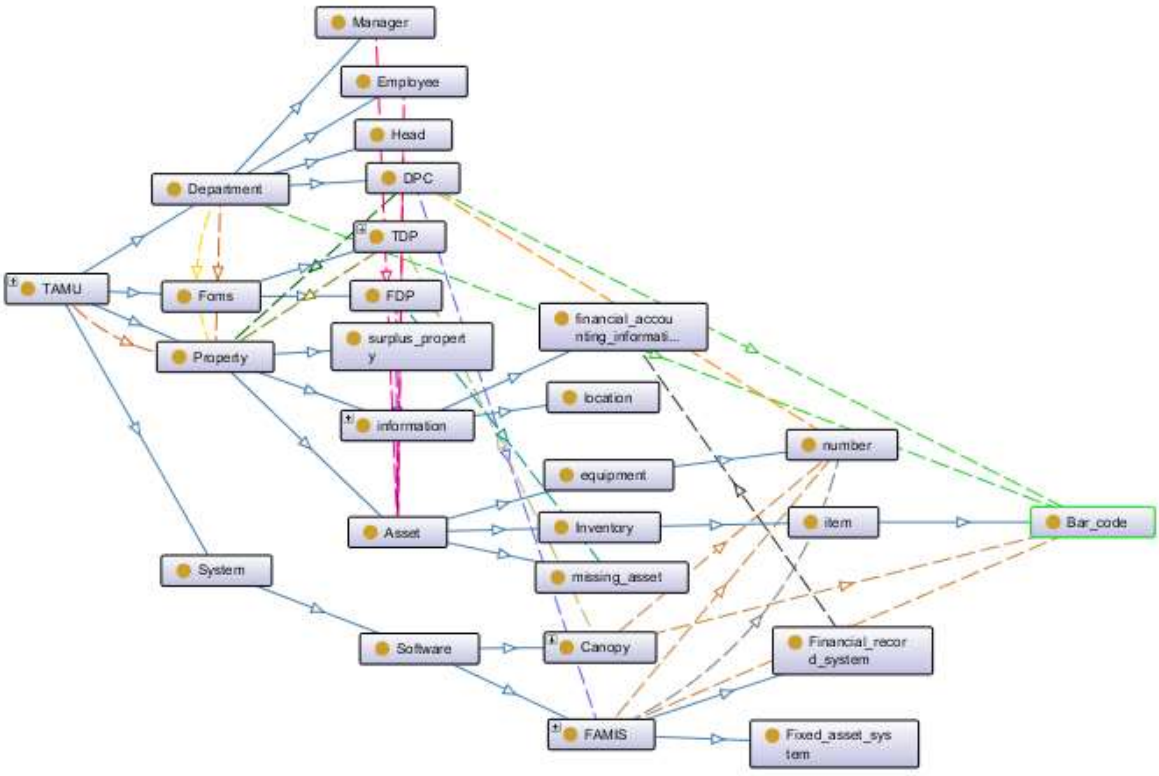


Figure 20: Final knowledge map from Protégé

Figure 20 shows the final knowledge map created in Protégé. The lines indicate relations among the item, and different colors have different meanings.

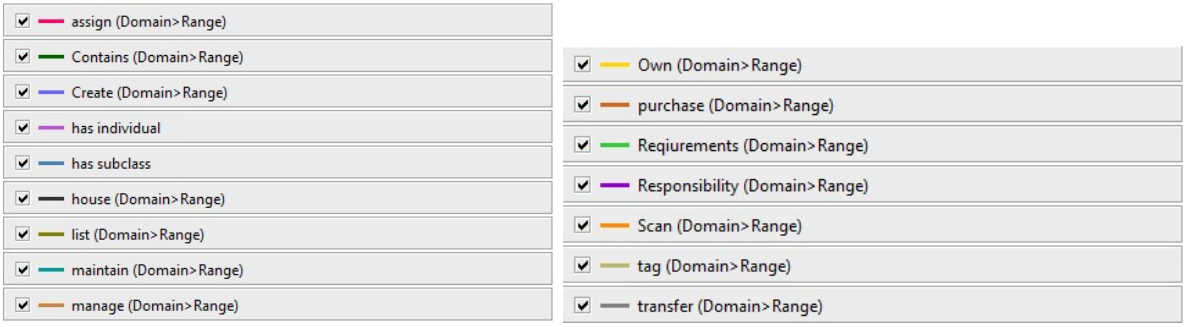


Figure 21: Relations using lines with different colors

Figure 21 shows the different meanings of the different colored lines.

6.4 Evaluation

The created knowledge map was evaluated by the Property Management unit of Financial Management Operations that wrote and maintains the manuals. Evaluation consists of three parts, integrity, correctness, and readability of the map. It was found that the relations in the map are accurate, and suggest two instances, equipment and manager, where the label content could be expanded to better reflect what a user would be looking for. It was also found that the map is hard to follow because “there are numerous lines making it hard to read”. This issue can be improved by simplifying the map.

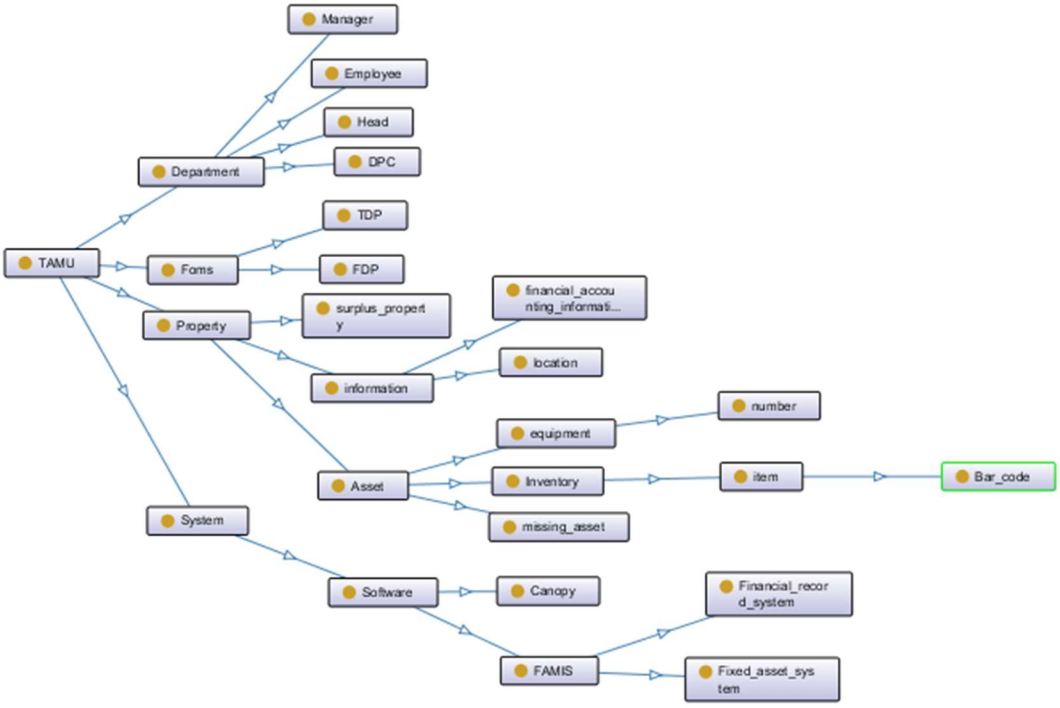


Figure 22: Simplified knowledge map

Figure 22 shows the simplified map. It only shows the relation of subclass and hids other relations. However, this might compromise the integrity of the map.

7. CONCLUSION AND DISCUSSION

This work presents a method for creating a knowledge map from unstructured text documents. The method first used NLP technologies to analyze unstructured text documents and then used Protégé to create the knowledge map. The created knowledge map was evaluated by the Property Management unit of Financial Management Operations that takes charge of those documents. The evaluation results suggest that even though the presented process can find most relations correctly, as the keywords are identified based on their frequencies, some important items might be missed. The map can also be simplified to increase readability, but this might negatively impact the integrity of the map.

This research presents a way for creating a knowledge map for FM from text documents. FM as an information-intensive industry, needs to deal with large amounts of information and different formats of data, creating problems such as interoperability. The industrial standards created to solve this interoperability problem cannot meet the unique requirements of various organizations. The proposed method can help facilities managers create a knowledge map using the documents within an organization, so that the uniqueness of each organization can be reflected in the structure of the map. The knowledge map can work as a visual aid to show where knowledge should be kept and can be found within the group or organization, as well as how to find those with the most expertise. When dealing with large amounts of complex data, not everyone is an expert on every subject. The map can lead people to the information and thus saves time and increases efficiency.

The knowledge map can be further developed as an expert system that can emulate the decision-making ability of a human expert. The knowledge map represents facts that can be used by the

inference engine. The inference engine reasons through the knowledge map and applies the rules to the known facts to deduce new facts and solve complex problems. For example, if there is some conflict information within the FAMIS system, the expert system can find the root cause of the problem in based on the relations in the map.

This research adopted the FP-growth to find relations between knowledge items in the domain of facilities management and is proven to be efficient. FP-growth has been applied to mining data in various domains such as medicine and biology, which has intensive information. However, it is seldom used in the domain of facilities management. This research can be used as a guidance for future studies in similar topics.

7.1 Study Limitations and Future Research

This proposed method suffers from the following limitations:

a. As the keywords are identified based on their frequencies, some important items with few frequencies may not be identified, meaning these are not reflected on the map.

b. As the articles are tokenized into sentences and words, the connection between sentences are broken. The meanings of pronouns such as it, this, and those are lost, and the pronouns are filtered out. The relations existing in the terms referred by pronouns may be lost.

c. The algorithm cannot read PDFs created by scanning. Therefore it loses some information when creating the map when dealing with pictures or scanned images.

d. The algorithm cannot self-validate for any missing, duplicated or inconsistent information. Thus, the finished knowledge map was reviewed by experts for its accuracy and completeness.

Suggested future work includes the following areas:

The algorithm used in the research adopts the part-of-speech tagging for labeling tokens, which means that it can only group the words by nouns, verbs, and prepositions. A more advanced tagging method such as Named Entity Reorganization Annotation, which can analyze the location and organization name, could improve the accuracy and efficiency of the mapping process.

Keywords are identified based on their frequencies under the assumption that the more important the word is, the more times it appears. This assumption may not apply to all terms in the documents. Future study can focus on other more "intelligent" ways to identify keywords.

Future studies can also be performed on the visualization effects of the map. For instance, the size of items and the lines between items can be adjusted based on their frequencies. The adjusted size might be more helpful for people to understand the map.

In the future, the next suggest task could be the application of the knowledge map in real daily operations such as developing an expert system for FM activities based on the knowledge map.

REFERENCES

Abdullah, S. A., Sulaiman, N., Ahmad Latiffi, A., and David, B. (2013). "Integration of facilities management (FM) practices with building information modeling (BIM)."

Allen, J. F. (2003). "Natural language processing."

Alouini, M.-S., and Goldsmith, A. J. (1999). "A unified approach for calculating error rates of linearly modulated signals over generalized fading channels." *IEEE Transactions on Communications*, 47(9), 1324-1334.

Anderson, A., Marsters, A., Dossick, C. S., and Neff, G. "Construction to operations exchange: Challenges of implementing COBie and BIM in a large owner organization." *Proc., Construction Research Congress 2012: Construction Challenges in a Flat World*, 688-697.

Aranda-Mena, G., and Wakefield, R. "Interoperability of building information-Myth of reality." *Proc., Proc. of the European Conference on Product and Process Modeling (ECPPM'2006)*, 127-134.

Araszkiewicz, K. (2017). "Digital technologies in Facility Management–the state of practice and research challenges." *Procedia Engineering*, 196, 1034-1042.

Baader, F., Horrocks, I., and Sattler, U. (2005). "Description logics as ontology languages for the semantic web." *Mechanizing Mathematical Reasoning*, Springer, 228-248.

Becerik-Gerber, B., Jazizadeh, F., Li, N., and Calis, G. (2011). "Application areas and data requirements for BIM-enabled facilities management." *Journal of construction engineering and management*, 138(3), 431-442.

Bikel, D. M., and Marcus, M. P. (2004). *On the parameter space of generative lexicalized statistical parsing models*, University of Pennsylvania.

Bird, S., Klein, E., and Loper, E. (2009). *Natural language processing with Python: analyzing text with the natural language toolkit*, " O'Reilly Media, Inc.".

Bird, S., Klein, E., Loper, E., and Baldridge, J. "Multidisciplinary instruction with the natural language toolkit." *Proc., Proceedings of the Third Workshop on Issues in Teaching Computational Linguistics*, Association for Computational Linguistics, 62-70.

Bird, S., and Loper, E. "NLTK: the natural language toolkit." *Proc., Proceedings of the ACL 2004 on Interactive poster and demonstration sessions*, Association for Computational Linguistics, 31.

Borgelt, C. "An Implementation of the FP-growth Algorithm." *Proc., Proceedings of the 1st international workshop on open source data mining: frequent pattern mining implementations*, ACM, 1-5.

Borst, W. N. (1997). *Construction of engineering ontologies for knowledge sharing and reuse*, Universiteit Twente.

Broder, A. Z., Glassman, S. C., Manasse, M. S., and Zweig, G. (1997). "Syntactic clustering of the web." *Computer Networks and ISDN Systems*, 29(8-13), 1157-1166.

Chen, G., and Luo, Y. "A BIM and ontology-based intelligent application framework." *Proc., Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC), 2016 IEEE*, IEEE, 494-497.

Chowdhury, G. G. (2003). "Natural language processing." *Annual review of information science and technology*, 37(1), 51-89.

Corcho, O., Fernández-López, M., and Gómez-Pérez, A. (2003). "Methodologies, tools and languages for building ontologies. Where is their meeting point?" *Data & knowledge engineering*, 46(1), 41-64.

Drachen, A. (2012). "Frequent Itemset and Association Rule Mining." <https://gameanalytics.com/blog/frequent-itemset-and-association-rule-mining-or-how-to-know-if-shirts-follows-pants-or-the-other-way-around.html>. (5/14, 2018).

East, E. W. (2007). "Construction operations building information exchange (Cobie): Requirements definition and pilot implementation standard." Engineer Research and Development Center, Champaign, IL. Construction Engineering Research Lab.

Farias, M., Roxin, A., and Nicolle, C. "Cobieowl, an owl ontology based on cobie standard." *Proc., OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"*, Springer, 361-377.

Fong, A. C. M., Hui, S. C., and Lau, C. (2004). "On-demand learning for a wireless campus." *IEEE MultiMedia*, 11(4), 50-60.

Gaines, B. R., and Shaw, M. L. "Collaboration through concept maps." *Proc., The first international conference on Computer support for collaborative learning*, L. Erlbaum Associates Inc., 135-138.

GCR, N. (2004). "Cost analysis of inadequate interoperability in the US capital facilities industry." *National Institute of Standards and Technology (NIST)*.

Giaretta, P., and Guarino, N. (1995). "Ontologies and knowledge bases towards a terminological clarification." *Towards very large knowledge bases: knowledge building & knowledge sharing*, 25, 32.

Golabchi, A., and Kamat, V. R. (2013). "Evaluation of industry foundation classes for practical building information modeling interoperability." *Ann Arbor*, 1001, 48109.

Guarino, N. "Formal ontology and information systems." *Proc., Proceedings of FOIS*, 81-97.

Halfawy, M. M., and Froese, T. M. (2007). "Component-based framework for implementing integrated architectural/engineering/construction project systems." *Journal of Computing in Civil Engineering*, 21(6), 441-452.

Han, J., Pei, J., and Kamber, M. (2011). *Data mining: concepts and techniques*, Elsevier.

Han, J., Pei, J., and Yin, Y. "Mining frequent patterns without candidate generation." *Proc., ACM sigmod record*, ACM, 1-12.

Horrocks, I., and Patel-Schneider, P. (2004). "Reducing OWL entailment to description logic satisfiability." *Web Semantics: Science, Services and Agents on the World Wide Web*, 1(4), 345-357.

IFMA (2009). "What is Facility Management?" *4/14*.

Kang, T. W., and Hong, C. H. (2015). "A study on software architecture for effective BIM/GIS-based facility management data integration." *Automation in Construction*, 54, 25-38.

Karola, A., Lahtela, H., Hänninen, R., Hitchcock, R., Chen, Q., Dajka, S., and Hagström, K. (2002). "BSPro COM-Server—interoperability between software tools using industrial foundation classes." *Energy and buildings*, 34(9), 901-907.

Karshenas, S., and Niknam, M. (2013). "Ontology-based building information modeling." *Computing in Civil Engineering (2013)*, 476-483.

Khan, A., Baharudin, B., Lee, L. H., and Khan, K. (2010). "A review of machine learning algorithms for text-documents classification." *Journal of advances in information technology*, 1(1), 4-20.

Kim, H.-G., Fillies, C., Smith, B., and Wikarski, D. (2002). "Visualizing a dynamic knowledge map using semantic web technology." *Engineering and Deployment of Cooperative Information Systems*, Springer, 130-140.

Kim, K., Kim, H., Kim, W., Kim, C., Kim, J., and Yu, J. (2018). "Integration of ifc objects and facility management work information using Semantic Web." *Automation in Construction*, 87, 173-187.

Kuda, F., and Berankova, E. "Integration of facility management and project management as an effective management tool for development projects." *Proc., Applied Mechanics and Materials*, Trans Tech Publ, 2676-2681.

Langendoen, D. T. (1993). "Natural Language and Universal Grammar." JSTOR.

Leinhardt, G. (1987). "Development of an expert explanation: An analysis of a sequence of subtraction lessons." *Cognition and instruction*, 4(4), 225-282.

Li, H., Wang, Y., Zhang, D., Zhang, M., and Chang, E. Y. "Pfp: parallel fp-growth for query recommendation." *Proc., Proceedings of the 2008 ACM conference on Recommender systems*, ACM, 107-114.

Liddy, E. D. (2001). "Natural language processing."

Loper, E., and Bird, S. "NLTK: The natural language toolkit." *Proc., Proceedings of the ACL-02 Workshop on Effective tools and methodologies for teaching natural language processing and computational linguistics-Volume 1*, Association for Computational Linguistics, 63-70.

Madnani, N. (2007). "Getting started on natural language processing with Python." *Crossroads*, 13(4), 5-5.

Manning, C. D., Raghavan, P., and Schütze, H. (2008). *Introduction to information retrieval*, Cambridge university press Cambridge.

Mckinstry (2013). "Total Cost of Ownership." The Builders' Association.

Mignard, C., and Nicolle, C. (2014). "Merging BIM and GIS using ontologies application to urban facility management in ACTIVe3D." *Computers in Industry*, 65(9), 1276-1290.

Mindmanager (2018). "What is a Knowledge Map?", <https://www.mindjet.com/features/knowledge-map/>. (5/24, 2018).

Mishra, M., and Vishwakarma, S. K. (2017). "Text Classification based on Association Rule Mining Technique." *International Journal of Computer Applications* 169(10).

mstamy2 (2011). "PyPDF2." <https://github.com/mstamy2/PyPDF2>. (5/15, 2018).

Nenonen, S., Jensen, P. A., and Lindahl, G. (2014). "Knowledge map of facilities management." *ING IN*, 247.

Niskanen, I., Purhonen, A., and Kuusijärvi, J. "Towards Semantic Facility Data Management." *Proc., The Third International Conference on Intelligent Systems and Applications*.

Parsanezhad, P., and Dimyadi, J. (2013). "Effective facility management and operations via a BIM-based integrated information system."

Pittet, P., Cruz, C., and Nicolle, C. (2014). "An ontology change management approach for facility management." *Computers in Industry*, 65(9), 1301-1315.

Pustejovsky, J., and Stubbs, A. (2012). *Natural Language Annotation for Machine Learning: A guide to corpus-building for applications*, " O'Reilly Media, Inc.".

Sabol, L. (2008). "Building information modeling & facility management." *IFMA World workplace*, 2-13.

Sagonas, K. (2006). "Association Rules & Frequent Itemsets." <http://user.it.uu.se/~kostis/Teaching/DM-05/Slides/association1.pdf>. (6/5, 2018).

SAS (2017). "Natural Language Processing: What it is and why it matters." <https://www.sas.com/en_us/insights/analytics/what-is-natural-language-processing-nlp.html>. (3/19, 2018).

Shen, W., Hao, Q., Mak, H., Neelamkavil, J., Xie, H., Dickinson, J., Thomas, R., Pardasani, A., and Xue, H. (2010). "Systems integration and collaboration in architecture, engineering, construction, and facilities management: A review." *Advanced engineering informatics*, 24(2), 196-207.

Smith, B., and Welty, C. "Ontology: Towards a new synthesis." *Proc., Formal Ontology in Information Systems*, ACM Press, USA, pp. iii-x, 3-9.

Smith, D. K., and Tardif, M. (2009). *Building information modeling: a strategic implementation guide for architects, engineers, constructors, and real estate asset managers*, John Wiley & Sons.

Sugumaran, V., and Storey, V. C. (2002). "Ontologies for conceptual modeling: their creation, use, and management." *Data & knowledge engineering*, 42(3), 251-271.

Tari, F. (2008). "A Review of Emerging Technological Trends in E-Learning." *Handbook of Research on Instructional Systems and Technology*, IGI Global, 729-740.

Taye, M. M. (2011). "Web-Based Ontology Languages and its Based Description Logics." *The Research Bulletin of Jordan ACM*, 2, 1-9.

TextMiner (2014). "Dive Into NLTK, Part II: Sentence Tokenize and Word Tokenize." <http://textminingonline.com/dive-into-nltk-part-ii-sentence-tokenize-and-word-tokenize>. (5/16, 2018).

Toledo-Alvarado, J., Guzman-Arenas, A., and Martínez-Luna, G. (2012). "Automatic building of an ontology from a corpus of text documents using data mining tools." *Journal of applied research and technology*, 10(3), 398-404.

Tomašević, N. M., Batić, M. Č., Blanes, L. M., Keane, M. M., and Vraneš, S. (2015). "Ontology-based facility data model for energy management." *Advanced Engineering Informatics*, 29(4), 971-984.

Tserng, H. P., Yin, S. Y. L., and Lee, M. H. (2010). "The use of knowledge map model in construction industry." *Journal of Civil Engineering and Management*, 16(3), 332-344.

University, S. (2016). "Protege." <https://protege.stanford.edu/products.php>. (5/13, 2018).

Uschold, M., and King, M. (1995). "Towards a methodology for building ontologies."

van Atteveldt, W. (2001). "Semantic Clarity or Universal Expressiveness?".

Vanlande, R., Nicolle, C., and Cruz, C. (2008). "IFC and building lifecycle management." *Automation in construction*, 18(1), 70-78.

Webster, J. J., and Kit, C. "Tokenization as the initial phase in NLP." *Proc., Proceedings of the 14th conference on Computational linguistics-Volume 4*, Association for Computational Linguistics, 1106-1110.

APPENDIX A

THE ALGORITHM USED FOR NATURAL LANGUAGE PROCESS AND FP MINING

```
import PyPDF2
from nltk.tokenize import word_tokenize
from nltk.tokenize import sent_tokenize
from nltk.stem import PorterStemmer
from collections import Counter
from nltk.corpus import stopwords
import re
from tika import parser
```

```
with open('procedures_manual.pdf','rb') as pdf_file, open('a.txt', 'w') as text_file:
    read_pdf = PyPDF2.PdfFileReader(pdf_file)
    number_of_pages = read_pdf.getNumPages()
    for page_number in range(96,115):   # use xrange in Py2
        page = read_pdf.getPage(page_number)
        page_content = page.extractText()
        text_file.write(page_content)
```

```
file_content = open("a.txt").read()
```

```
print (file_content)
```

```
tokens = word_tokenize(file_content)
sentences = sent_tokenize(file_content)
```

```
print (tokens)

Counter(tokens).most_common()[:100]
```

```
from nltk import collocations
bigram_measures = collocations.BigramAssocMeasures()
bigram_finder = collocations.BigramCollocationFinder.from_words(tokens)

# Filter to top 20 results; otherwise this will take a LONG time to analyze
bigram_finder.apply_freq_filter(20)
for bigram in bigram_finder.score_ngrams(bigram_measures.raw_freq)[:10]:
 print (bigram)
```

```
token = re.findall(r'A&M|Texas A&M|Property Management|Procedures Manual|Procedures Manager\S+',file_content)
```

```
from nltk import collocations
bigram_measures = collocations.BigramAssocMeasures()
bigram_finder = collocations.BigramCollocationFinder.from_words(tokens)

# Filter to top 20 results; otherwise this will take a LONG time to analyze
bigram_finder.apply_freq_filter(20)
for bigram in bigram_finder.score_ngrams(bigram_measures.raw_freq)[:10]:
 print (bigram)
```

```
stop_words = set(stopwords.words("english"))
```

```
print (stop_words)
```

```
stop_words.add('The')
stop_words.add(',')
stop_words.add('-')
stop_words.add('.')
stop_words.add('2016')
stop_words.add('Feburary')
stop_words.add('1')
stop_words.add('R')
stop_words.add('421')
stop_words.add('if')
stop_words.add('(')
stop_words.add(')')
stop_words.add('number')
```

```
print (token)
```

```
filtered_words = []
for w in token:
    if w not in stop_words:
        filtered_words.append(w)
print (filtered_words)
```

```
sentences = sent_tokenize(file_content)
search_keywords=['location','software','Software','System','Transfering''Surplus','surplus','transfer','Transfer','Trans
for sentence in sentences:
    lst = []
    for word in search_keywords:
        if word in sentence:
            lst.append(word)
    print('{0} key word(s) in sentence: {1}'.format(len(lst), ', '.join(lst)))
    print(sentence + "\n")
```

```
class treeNode:
    def __init__(self, nameValue, numOccur, parentNode):
        self.name = nameValue
        self.count = numOccur
        self.nodeLink = None
        self.parent = parentNode      #needs to be updated
        self.children = {}
    def inc(self, numOccur):
        self.count += numOccur
    def disp(self, ind=1):
        print ('  '*ind, self.name, ' ', self.count)
        for child in self.children.values():
            child.disp(ind+1)
```

```
def createTree(dataSet, minSup=1): #create FP-tree from dataset but don't mine
    headerTable = {}
    #go over dataSet twice
    for trans in dataSet:#first pass counts frequency of occurance
        for item in trans:
            headerTable[item] = headerTable.get(item, 0) + dataSet[trans]
    for k in list(headerTable):  #remove items not meeting minSup
        if headerTable[k] < minSup:
            del(headerTable[k])
    freqItemSet = set(headerTable.keys())
    #print 'freqItemSet: ',freqItemSet
    if len(freqItemSet) == 0: return None, None  #if no items meet min support -->get out
    for k in headerTable:
        headerTable[k] = [headerTable[k], None] #reformat headerTable to use Node link
    #print 'headerTable: ',headerTable
    retTree = treeNode('Null Set', 1, None) #create tree
    for tranSet, count in dataSet.items():  #go through dataset 2nd time
        localD = {}
        for item in tranSet:  #put transaction items in order
            if item in freqItemSet:
                localD[item] = headerTable[item][0]
        if len(localD) > 0:
            orderedItems = [v[0] for v in sorted(localD.items(), key=lambda p: p[1], reverse=True)]
            updateTree(orderedItems, retTree, headerTable, count)#populate tree with ordered freq itemset
    return retTree, headerTable #return tree and header table
```

```
def updateTree(items, inTree, headerTable, count):
    if items[0] in inTree.children:#check if orderedItems[0] in retTree.children
        inTree.children[items[0]].inc(count) #incrament count
    else:   #add items[0] to inTree.children
        inTree.children[items[0]] = treeNode(items[0], count, inTree)
        if headerTable[items[0]][1] == None: #update header table
            headerTable[items[0]][1] = inTree.children[items[0]]
        else:
            updateHeader(headerTable[items[0]][1], inTree.children[items[0]])
    if len(items) > 1:#call updateTree() with remaining ordered items
        updateTree(items[1::], inTree.children[items[0]], headerTable, count)
```

```
def updateHeader(nodeToTest, targetNode):   #this version does not use recursion
    while (nodeToTest.nodeLink != None):    #Do not use recursion to traverse a linked list!
        nodeToTest = nodeToTest.nodeLink
    nodeToTest.nodeLink = targetNode
```

```
def loadSimpDat():
    simpDat = [


               ]

    return simpDat
```

```
def createInitSet(dataSet):
    retDict = {}
    for trans in dataSet:
        retDict[frozenset(trans)] = 1
    return retDict
```

```
simpDat = loadSimpDat()
```

```
simpDat
```

```
def createInitSet(dataSet):
    retDict = {}
    for trans in dataSet:
        retDict[frozenset(trans)] = 1
    return retDict
```

```
initSet = createInitSet(simpDat)
```

```
initSet
```

```
#The FP-tree
myFPtree, myHeaderTab = createTree(initSet, 0)
```

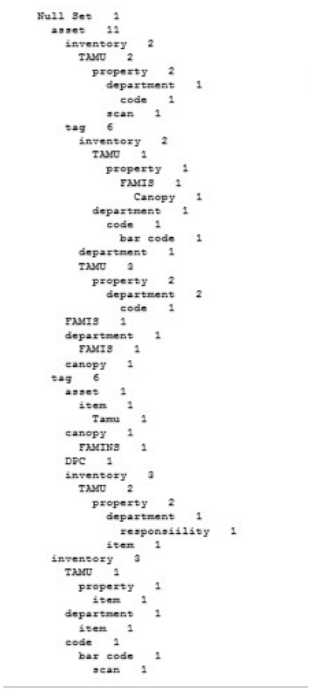
```
myFPtree.disp()
```

APPENDIX B

THE RESULTS OF FP MINING

```
Null Set  1
  asset  25
    inventory  12
      property  6
        TAMU  5
          item  2
            canopy  1
          scan  1
          code  2
            FAMIS  1
              department  1
                canopy  1
            item  1
      item  2
        canopy  1
      FAMIS  1
      code  1
        item  1
      TAMU  1
        code  1
    item  1
    code  2
      item  1
    property  5
      FAMIS  1
      TAMU  3
        FAMIS  1
          department  1
            canopy  1
        code  1
          FAMIS  1
            DPC  1
              department  1
        code  1
    canopy  1
    DPC  1
    FAMIS  2
      DPC  1
      code  1
    department  1
      DPC  1
        prooperty  1
  FAMIS  2
    item  1
      Canopy  1
    DPC  1
  inventory  6
    property  1
      TAMU  1
        item  1
    FAMIS  3
      TAMU  1
      department  1
      code  1
        canopy  1
    item  1
    code  1
  property  1
    TAMU  1
      department  1
  code  1
```

```
[['item', 'TAMU', 'property', 'asset', 'inventory'],
 ['Canopy', 'item', 'FAMIS'],
 ['item', 'asset'],
 ['item', 'asset', 'inventory'],
 ['property', 'asset', 'inventory'],
 ['item', 'TAMU', 'property', 'inventory'],
 ['TAMU', 'property', 'asset', 'inventory', 'scan'],
 ['asset', 'inventory'],
 ['code', 'asset'],
 ['FAMIS', 'property', 'asset'],
 ['TAMU', 'FAMIS', 'inventory'],
 ['canopy',
  'TAMU',
  'code',
  'FAMIS',
  'property',
  'asset',
  'inventory',
  'department'],
 ['FAMIS', 'DPC'],
 ['FAMIS', 'inventory', 'department'],
 ['canopy', 'asset'],
 ['item', 'code', 'TAMU', 'property', 'asset', 'inventory'],
 ['DPC', 'asset'],
 ['FAMIS', 'asset', 'inventory'],
 ['canopy', 'TAMU', 'FAMIS', 'property', 'asset', 'department'],
 ['TAMU', 'property', 'asset'],
 ['FAMIS', 'DPC', 'asset'],
 ['canopy', 'item', 'TAMU', 'property', 'asset', 'inventory'],
 ['TAMU', 'property', 'department'],
 ['item', 'inventory'],
 ['canopy', 'item', 'asset', 'inventory'],
 ['item', 'code', 'asset'],
 ['FAMIS', 'property', 'asset'],
 ['item', 'code', 'asset', 'inventory'],
 ['code', 'inventory'],
 ['canopy', 'code', 'FAMIS', 'inventory'],
 ['code', 'property', 'asset'],
 ['code', 'TAMU', 'asset', 'inventory'],
 ['code', 'FAMIS', 'asset'],
 ['code', 'TAMU', 'FAMIS', 'property', 'DPC', 'asset', 'department'],
 ['code', 'DPC'],
 ['prooperty', 'DPC', 'asset', 'department'],
 ['code', 'DPC']]
```



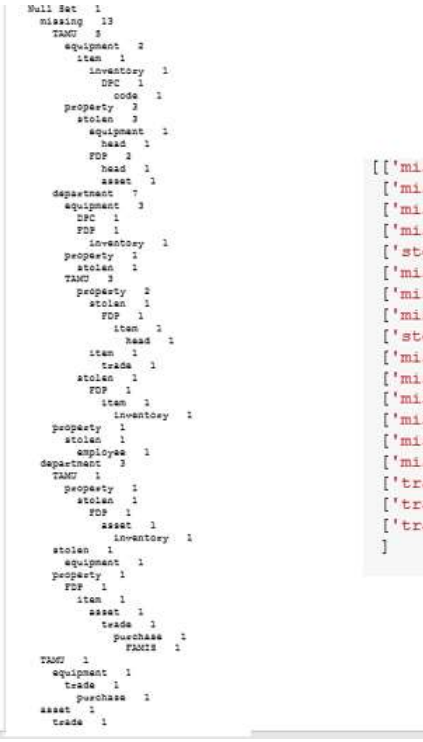
```
Null Set  1
  asset  11
    inventory  2
      TAMU  2
        property  2
          department  1
            code  1
          scan  1
    tag  6
      inventory  2
        TAMU  1
          property  1
            FAMIS  1
              Canopy  1
        department  1
          code  1
            bar code  1
      department  1
      TAMU  3
        property  2
          department  2
            code  1
    FAMIS  1
    department  1
      FAMIS  1
    canopy  1
  tag  6
    asset  1
      item  1
        Tamu  1
    canopy  1
      FAMINS  1
    DPC  1
    inventory  3
      TAMU  2
        property  2
          department  1
            responsiility  1
          item  1
  inventory  3
    TAMU  1
      property  1
        item  1
    department  1
      item  1
    code  1
      bar code  1
        scan  1
```

```
[['code', 'TAMU', 'property', 'asset', 'inventory', 'department'],
 ['tag', 'Canopy', 'TAMU', 'FAMIS', 'property', 'asset', 'inventory'],
 ['tag', 'asset', 'department'],
 ['tag', 'item', 'Tamu', 'asset'],
 ['FAMIS', 'asset'],
 ['item', 'TAMU', 'property', 'inventory'],
 ['TAMU', 'property', 'asset', 'inventory', 'scan'],
 ['tag', 'canopy', 'FAMINS'],
 ['tag', 'DPC'],
 ['FAMIS', 'asset', 'department'],
 ['tag', 'inventory'],
 ['tag', 'code', 'TAMU', 'property', 'asset', 'department'],
 ['tag', 'TAMU', 'property', 'asset', 'department'],
 ['tag', 'bar code', 'code', 'asset', 'inventory', 'department'],
 ['item', 'inventory', 'department'],
 ['canopy', 'asset'],
 ['tag', 'TAMU', 'responsiility', 'property', 'inventory', 'department'],
 ['tag', 'TAMU', 'asset'],
 ['item', 'TAMU', 'property', 'inventory'],
 ['bar code', 'code', 'inventory', 'scan'],
 ['tag', 'TAMU', 'asset'],
 ['tag', 'code', 'TAMU', 'property', 'asset', 'department'],
 ['tag', 'item', 'TAMU', 'property', 'inventory'],
 ['item', 'TAMU', 'property', 'inventory']]
```

```
Null Set  1
  asset  23
    property  14
      department  2
        FAMIS  1
        tag  1
          equipment  1
            DPC  1
      Canopy  1
      TAMU  5
        department  1
          inventory  1
        purchase  3
          item  2
            FDP  1
              equipment  1
            tag  1
          FAMIS  1
            canopy  1
        equipment  1
        tag  1
          FDP  1
            equipment  1
        inventory  1
        FAMIS  1
          tag  1
      code  1
      tag  1
      FAMIS  1
        canopy  1
    department  2
      purchase  1
      inventory  1
    code  1
      purchase  1
    FAMIS  2
      code  1
      item  1
        DPC  1
    TAMU  3
      code  1
        FDP  1
          equipment  1
            property  1
        inventory  1
          tag  1
        canopy  1
      inventory  1
        FAMIS  1
          code  1
  property  7
    TAMU  6
      FAMIS  1
        FDP  1
      employee  1
      inventory  1
      department  2
        code  1
          inventory  1
            item  1
              bar code  1
        inventory  1
          FDP  1
      item  1
    item  1
  tag  1
    canopy  1
      FAMINS  1
```

```
      code  1
  property  7
    TAMU  6
      FAMIS  1
        FDP  1
      employee  1
      inventory  1
      department  2
        code  1
          inventory  1
            item  1
              bar code  1
        inventory  1
          FDP  1
      item  1
    item  1
  tag  1
    canopy  1
      FAMINS  1
  department  3
    equipment  1
    code  2
      purchase  2
        item  1
        equipment  1
  inventory  1
    item  1
      FDP  1
        equipment  1
          departmentt  1
  TAMU  1
    department  1
      FAMIS  1
        purchase  1
  code  1
    purchase  1
      assetinventory  1
```

```
[['FAMIS', 'property', 'asset', 'department'],
 ['purchase', 'asset', 'department'],
 ['asset', 'inventory', 'department'],
 ['TAMU', 'FAMIS', 'property', 'FDP'],
 ['purchase', 'code', 'asset'],
 ['Canopy', 'property', 'asset'],
 ['TAMU', 'property', 'asset', 'inventory', 'department'],
 ['tag', 'canopy', 'FAMINS'],
 ['code', 'FAMIS', 'asset'],
 ['TAMU', 'employee', 'property'],
 ['TAMU', 'property', 'inventory'],
 ['purchase', 'equipment', 'item', 'TAMU', 'property', 'FDP', 'asset'],
 ['equimpment', 'TAMU', 'property', 'asset'],
 ['equipment', 'tag', 'TAMU', 'property', 'FDP', 'asset'],
 ['equipmen', 'department'],
 ['item', 'property'],
 ['TAMU', 'property', 'asset', 'inventory'],
 ['code', 'property', 'asset'],
 ['purchase', 'item', 'code', 'department'],
 ['tag', 'property', 'asset'],
 ['equipment', 'code', 'TAMU', 'proerty', 'FDP', 'asset'],
 ['equipment', 'tag', 'property', 'DPC', 'asset', 'department'],
 ['purchase', 'equipment', 'code', 'department'],
 ['purchase', 'tag', 'item', 'TAMU', 'property', 'asset'],
 ['bar code', 'item', 'code', 'TAMU', 'property', 'inventory', 'department'],
 ['TAMU', 'property', 'FDP', 'inventory', 'department'],
 ['item', 'TAMU', 'property'],
 ['equipment', 'item', 'FDP', 'inventory', 'departmentt'],
 ['TAMU', 'property', 'asset', 'inventory', 'department'],
 ['tag', 'TAMU', 'asset', 'inventory'],
 ['tag', 'TAMU', 'FAMIS', 'property', 'asset'],
 ['purchase', 'TAMU', 'FAMIS', 'department'],
 ['canopy', 'TAMU', 'asset'],
 ['purchase', 'code', 'assetinventory'],
 ['code', 'FAMIS', 'asset', 'inventory'],
 ['item', 'FAMIS', 'DPC', 'asset'],
 ['canopy', 'FAMIS', 'property', 'asset'],
 ['purchase', 'canopy', 'TAMU', 'FAMIS', 'property', 'asset']]
```

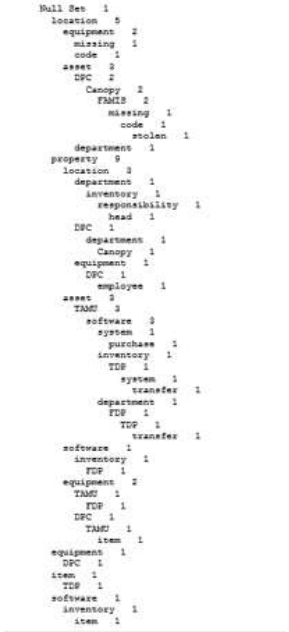


```
Null Set  1
  missing  13
    TAMU  5
      equipment  2
        item  1
          inventory  1
            DPC  1
              code  1
      property  2
        stolen  3
          equipment  1
            head  1
          FDP  2
            head  1
            asset  1
    department  7
      equipment  3
        DPC  1
        FDP  1
          inventory  1
      property  1
        stolen  1
      TAMU  3
        property  2
          stolen  1
            FDP  1
              item  1
                head  1
          item  1
            trade  1
        stolen  1
          FDP  1
            item  1
              inventory  1
    property  1
      stolen  1
        employee  1
  department  3
    TAMU  1
      property  1
        stolen  1
          FDP  1
            asset  1
              inventory  1
    stolen  1
      equipment  1
    property  1
      FDP  1
        item  1
          asset  1
            trade  1
              purchase  1
                FAMIS  1
  TAMU  1
    equipment  1
      trade  1
        purchase  1
  asset  1
    trade  1
```

```
[['missing','equipment','TAMU'],
 ['missing','equipment','department'],
 ['missing','stolen','property','department'],
 ['missing','stolen','employee','property'],
 ['stolen','TAMU','property','FDP','asset','inventory','department'],
 ['missing','stolen','equipment','TAMU','head','property'],
 ['missing','stolen','TAMU','head','property','FDP'],
 ['missing','stolen','item','TAMU','head','property','FDP','department'],
 ['stolen','equipment','department'],
 ['missing','equipment','DPC', 'department'],
 ['missing','equipment','item','code','TAMU','DPC','inventory'],
 ['missing','stolen','TAMU','property','FDP','asset'],
 ['missing','equipment','FDP','inventory','department'],
 ['missing','stolen','item','TAMU','FDP','inventory','department'],
 ['missing','trade','item','TAMU','property','department'],
 ['trade','purchase','equipment','TAMU'],
 ['trade','asset'],
 ['trade','purchase','item','FAMIS','property','FDP','asset','department']
 ]
```

```
Null Set  1
  transfer  20
    property  12
      equipment  1
      department  2
        equipment  1
          TDP  1
            FDP  1
              Canopy  1
        FDP  1
      TDP  2
        Canopy  1
        FAMIS  1
      TAMU  6
        TDP  1
        system  2
          surplus  1
            departm  1
          FDP  1
        equipment  2
          system  2
            FAMIS  1
              inventory  1
          department  1
            system  1
      item  1
        asset  1
          inventory  1
            purchase  1
    FDP  1
    department  1
      surplus  1
    TAMU  4
      department  2
        equipment  1
          TDP  1
        system  1
      asset  1
        code  1
          acquisition  1
      system  1
        asset  1
          code  1
            acquisition  1
    item  1
      asset  1
    equipment  1
      TDP  1
  equipment  2
    item  1
      invent  1
    TDP  1
      surplus  1
  TDP  1
    item  1
  property  4
    department  1
      item  1
        surplus  1
          tag  1
    TAMU  2
      item  1
        surplus  1
      department  2
        equipment  1
          FDP  1
        asset  1
          code  1
            tag  1
              bar code  1
  TAMU  1
    surplus  1
```

```
[['transfer', 'equipment', 'property'],
 ['transfer', 'FDP'],
 ['surplus', 'transfer', 'department'],
 ['transfer', 'equipment', 'TAMU', 'TDP', 'department'],
 ['transfer', 'item', 'asset'],
 ['transfer', 'equipment', 'Canopy', 'property', 'FDP', 'TDP', 'department'],
 ['system', 'transfer', 'TAMU', 'department'],
 ['equipment', 'item', 'invent'],
 ['transfer', 'Canopy', 'property', 'TDP'],
 ['transfer', 'TAMU', 'property', 'TDP'],
 ['system', 'surplus', 'transfer', 'TAMU', 'property', 'departm'],
 ['item', 'TDP'],
 ['transfer', 'purchase', 'item', 'property', 'asset', 'inventory'],
 ['surplus', 'tag', 'item', 'property', 'department'],
 ['surplus', 'item', 'TAMU', 'property'],
 ['surplus', 'equipment', 'TDP'],
 ['transfer', 'equipment', 'TDP'],
 ['system', 'transfer', 'equipment', 'TAMU', 'FAMIS', 'property'],
 ['transfer', 'FAMIS', 'property', 'TDP'],
 ['system', 'transfer', 'equipment', 'TAMU', 'property', 'inventory'],
 ['transfer', 'property', 'FDP', 'department'],
 ['transfer', 'acquisition', 'code', 'TAMU', 'asset'],
 ['system', 'transfer', 'TAMU', 'property', 'FDP'],
 ['system', 'transfer', 'TAMU', 'property', 'department'],
 ['surplus', 'TAMU'],
 ['equipment', 'TAMU', 'property', 'FDP', 'department'],
 ['tag', 'bar code', 'code', 'TAMU', 'property', 'asset', 'department'],
 ['system', 'transfer', 'acquisition', 'code', 'TAMU', 'asset']]
```



```
Null Set  1
  location  5
    equipment  2
      missing  1
        code  1
      asset  2
        DPC  2
          Canopy  2
            FAMIS  2
              missing  1
                code  1
                  stolen  1
        department  1
    property  9
      location  3
        department  1
          inventory  1
            responsibility  1
              head  1
        DPC  1
          department  1
            Canopy  1
        equipment  1
          DPC  1
            employee  1
      asset  3
        TAMU  3
          software  3
            system  1
              purchase  1
            inventory  1
              TDP  1
                system  1
                  transfer  1
              department  1
                FDP  1
                  TDP  1
                    transfer  1
      software  1
        inventory  1
          FDP  1
      equipment  2
        TAMU  1
          TDP  1
        DPC  1
          TAMU  1
            item  1
    equipment  1
      DPC  1
    item  1
      TDP  1
    software  1
      inventory  1
        item  1
```

```
[['location','missing','equipment'],
 ['location','responsibility','head','property','inventory','department'],
 ['location','Canopy','FAMIS','DPC','asset'],
 ['location','Canopy','property','DPC','department'],
 ['equipment','DPC'],
 ['location','equipment','code'],
 ['location','equipment','employee','property','DPC'],
 ['location','missing','stolen','Canopy','code','FAMIS','DPC','asset'],
 ['location','asset','department'],
 ['software','system','purchase','TAMU','property','asset'],

 ['software','system','transfer','TAMU','property','asset','inventory','TDP'],
 ['item','TDP'],
 ['software','item','inventory'],
 ['software','property','FDP','inventory'],
 ['software','transfer','TAMU','property','FDP','asset','TDP','department'],
 ['equipment','TAMU','property','FDP'],
 ['equipment','item','TAMU','property','DPC'],
```