Universiteit Leiden

ICT in Business

Comparing the Use of Technology Among Industry Branches

Name: Hanyu Zhu
Student-no: s1684736

Date: 15/08/2017

1st academic supervisor: Dr. Werner Heijstek
2nd academic supervisor: Dr. Enrique Larios Vargas

1st company supervisor: Drs. Xander Schrijen
2nd company supervisor: Drs. Dennis Bijlsma

MASTER'S THESIS

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Abstract

The technology selection is critical for the project success. However, the lack of validated guidelines makes it challenging to select the technologies for the system implementation. There are many factors that may affect the final technology selection, like the developer's preference, industry's preference, development costs, system's functional requirements and so on. This research targets the influence of the industry's preference on the technology selection. Through comparing the technology usage among the industry branches, the research addresses the question: To what extent do different industries make different technology decisions for implementing software systems? The data sets used for this research are collected from the Software Improvement Group (SIG)'s data warehouse.

The research is split into three steps. First, based on the interviews with 14 interviewees, a collection of 1,519 systems is categorized into 10 industry branches. Second, the used technologies for the industry comparison are collected from the systems. During this process, the *technology stacks* and the *abstract stacks* are created to collect the technology combinations and find the method to compare these technologies respectively. In the last step, the relation between the selected technologies and the industry branches is detected by visualizing the results from the previous two steps. Overall, there is no significant difference among different industry branches while selecting the technologies for the system implementation. Java together with Java-based technologies and C# together with C#-based technologies are the most popular technologies among all the industries. Generally, Java and Java-based technologies are used more frequently than C# and C#-based technologies, according to our data sets. However, compared with other industries, the Technology-Software & Computer Services industry is more in favor of ASP.NET. And moreover, this industry is the only one that use more T-SQL than PL/SQL.

Acknowledgement

This thesis is the final product for my two-year's master program as an ICT in Business student in Leiden University. The research is done in the Software Improvement Group (SIG) and led by four kind supervisors.

First, I want to express my gratitude to my company supervisors, Drs. Xander Schrijen and Drs. Dennis Bijlsma. They helped me to find an interesting research topic and gave me timely and valuable feedbacks as well as suggestions throughout these seven months. This thesis cannot be finished without their guidelines in every single step of the research.

Second, I am going to appreciate my academic supervisors, Dr. Werner Heijstek and Dr. Enrique Larios Vargas. They coached me to do a scientific research and reviewed my thesis from the academic perspective to make it valuable to the scientific community.

And then I want to praise all the people in SIG for the friendly support during my internship. I learnt a lot from these awesome people.

At last, I would like to thank my parents for their endless love. I also want to thank my friends for sharing their experience during the research with me and spending many happy times together with me to extricate me from the stress.

Table of Contents

Chapter 1 Introduction

1.1 Problem Statement & Research Motivation

Information systems projects frequently fail. Studies have shown that the failure rate of large projects is between 50% -80% (Dorsey, 2005). According to Kaur et al. (2013), the Avanade Research Report in 2007 shows that 66% of the failure is due to system specification, 51% due to requirement understanding, and 49% due to technology selections. Besides, Mandal et al. (2015) list project failure reasons originating from technology sources as following: 1. Wrong technology selection; 2. Technology too new or didn't work as expected; 3. Use of immature technology; 4. Technology planning. Therefore, the technology selection appears to be a critical factor for project success.

Generally, during the system design and construction process, a software developer makes dozens of decisions. Sometimes this involves solving a problem unique to a particular domain space or a particular architectural issue. Other times it is about which technology is the best for a particular purpose. That is actually one of the most critical pieces of getting a project right (Hall, 2017). For instance, if an IT company chooses COBOL as its development language or banks on FoxPro as their database backend for new projects, it might have adverse results (Shojaee, 2007). Thus, it is generally believed that choosing the right technology really matters. Shojaee (2007) lists a number of interesting points to support it in his blog: 1. Choosing the right technology will make sure to attract the best possible talent; 2. The right technology will scale well as the application grows in popularity; 3. The right technology will make sure the execution speed; 4. It will make sure the code is easily maintained, enhanced and expanded; 5. Popular technologies are well supported by the industry and profitable companies. They are able to help system developers avoid future pitfalls.

However, there is little documentation available of how the technology integration can be accomplished (Bouwers, 2013). Thus, while selecting the technologies for the projects, the developers have few validated guidelines that can be referred to. The technology selection depends on many factors, like the developer's preference, industry's preference, development costs, system's functional requirements and so on. However, among these factors, which factors really affect the final technology selection?

To answer this question, we need to detect each factor's influence on the final technology selection. This research targets the influence of the industry's preference on the technology selection. Will different industries have different technology selections while implementing the systems? We are going to answer this question by doing this research. We come up with a main question which can be split into three sub research questions.

1.2 Research Questions

Main Research Question:
To what extent do different industries make different technology decisions for implementing software systems?

To answer this question, we need to detect the use of the technology for the system implementation in each industry branch. However, how to define the terms "*Technology*" and "*Industry Branch*" should be answered first.

Since the term "*Technology*" includes programming languages, build tools and runtime-components such as interpreters and servers (Bouwers, 2013), our research targets on the usage of general-purpose technologies and some domain-specific technologies like database technologies, user interface technologies and technologies that manage data exchange. This will be described in detail in Chapter 3.

A system is a set of interacting or interdependent component parts forming a complex or intricate whole (Merriam-Webster, 2017). Every system is delineated by its spatial and temporal boundaries, surrounded and influenced by its environment, described by its structure and purpose and expressed in its functioning. The "*Industry Branch*" of the system means the industrial source of the system. In other words, for the usage of which industry the system is created.

Moreover, since this research is focusing on the industry's technology preference, we are trying to mitigate the influence caused by other factors. In other words, we are going to find that whether there are some technologies that are only commonly used by some industries, but not frequently used by the others. The technologies that are used for only a few systems will be excluded from this research, in case these distinct technology selections are caused by the developer's or the project's preferences. To detect the relation between the industry's preference and the technology selection, we make an assumption that there are some technologies that are only widely used by some industries, but not frequently used by the others. Thus, the technologies that will be collected for the industry comparison should be relatively commonly used.

To get the answers to the main question, the research can be split into several steps. First, we need to collect the systems and find that in which industry branches are these systems distributed. Second, we will try to collect the technologies that are in the common uses from these systems. With this information, we are able to compare the use of these technologies among the industry branches. Based on these steps, the main research question is split into the following three sub research questions:
1. How to classify systems into corresponding industry branches?
2. Can we find commonly used technologies from these systems?
3. What is the relation between the results from sub research questions 1 and 2?

1.3 Research Context

SIG
Software Improvement Group. It was born in 2000 with the headquarters in Amsterdam, The Netherlands. It is a highly specialized consultancy company for quality of software, providing insight into the technical quality of software systems and advice on how to improve. This research is done with the help of people in SIG. Moreover, all the original data for the research is collected from SIG's data warehouse.

SAW
Software Analysis Warehouse, SIG's data warehouse which stores the persisted analysis results generated by Software Analysis Toolkit (SAT), a source code analysis tool developed and used

by SIG. The analysis results contain the information of the systems that have been analyzed by the analysts in SIG for its clients around the world. The information includes the system name, analyst name, analysis date, system implementation technology, technology volume, system maintainability ratios and so on.

Since the research is conducted by using R and Python, the terminology "*Data Frame*" is also used to represent the "*Data Set*" in this thesis.

1.4 Research Methodologies

Main Research Question:
To what extent do different industries make different technology decisions for implementing software systems?

1. How to classify systems into corresponding industry branches?
There is no industry branch information for the systems in SAW. Thus, to answer this question, we have to extract the existing system information from SAW, find a widely used industry classification benchmark and conduct the interviews with the system analysts in SIG to get the classification results.

2. Can we find commonly used technologies from these systems?
To answer this question, we have to find the methods to group these systems based on their technology uses and then detect the technologies from these groups for the further analysis. To group the systems, first, literature review will be conducted to find the existing steps and methods about how to group data from previous research. Then based on our research situation, we will write the most suitable algorithm to group the systems. After that, the commonly used technologies can be extracted from these groups.

3. What is the relation between the results from sub research questions 1 and 2?
To answer this question, we are going to visualize the results from sub research questions 1 and 2. By combining the system – industry categories and the technologies that are collected from the systems into one graph, it is easy to figure out whether there is a significant relation between the technology selection and the industry branch.

1.5 Thesis Structure

As it is shown in Figure 1.1, the structure of this thesis is built based on the sub research questions. Chapter 2 describes the benchmark we are going to use for the system - industry classification and the methods used for the categorization. The final system – industry categories will be described at the end of Chapter 2. In Chapter 3, we are going to collect the commonly used technologies from these systems. By grouping the systems based on their technology uses, the technologies for the industry comparison can be extracted from these groups. We will describe the methods for the grouping work, explain the algorithm we use and display the final results. In Chapter 4, we will describe the methods for detecting the relation between the system – industry categories and the collected technologies, and then display the main findings from the data sets. Finally, in the last chapter, we are going to make a conclusion for the whole research.

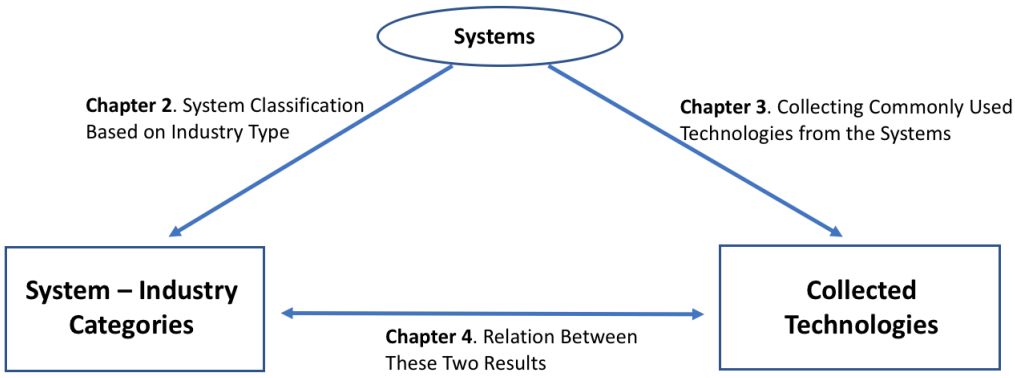


Figure 1.1 Thesis Structure

Chapter 2 System Classification Based on Industry Type

Currently, nearly all the companies implement their business processes through systems. In this chapter, we are going to classify these companies into industry types. There are several industry taxonomies widely used, like Global Industry Classification Standard (GICS), North American Industry Classification System (NAICS), Industry Classification Benchmark (ICB) and so on. The benchmark we use for the classification work is a modified version based on the Industry Classification Benchmark (FTSE Russell, 2012). It can be found in Appendix A. Compared with the others, this benchmark facilitates a clearer four-layer structure and provides detailed and comprehensive definitions for each sector. The systems information we are going to use for the classification is collected from SIG's data warehouse, Software Analysis Warehouse (SAW). However, there is no clear systems industry information in the warehouse. In this chapter, we describe the methodologies used to classify the systems into corresponding industry categories. We are going to extract the information like system name and system analysts' name from the data warehouse, find the interviewees and conduct the interviews with them. During this process, "*Double Checks*" and "*Hierarchical Classification*" are used to get the final categories. The systems distribution in each industry can be found at the end of this chapter.

2.1 Classification Methodology

In this part, the four methodologies used for the classification work are described.

Methodology 1: Data Extraction & Data Modelling

According to Levene et al. (2003), a data warehouse often integrates heterogeneous data from multiple and distributed information sources and contains historical and aggregated data. Data modelling is beneficial to view a data warehouse in terms of a dimensional model. The entity-relationship model can achieve a high degree of data independence and is based on set of theory and relation theory. It can be used as a basis for a unified view of data (Chen, 1976). Because SAW is a document-oriented database, unlike the rational databases that already organized data into one or more tables or relations, an entity-relationship model should be created before the data extraction from SAW. Figure 2.1 describes an entity-relationship model that contains two entity sets, "*Snapshots*" table and "*Analysts*" table as well as the binary relationships with 1: n mappings in which the existence of the n entities on the one side of the relationship depends on the existence of one entity on the other side of the relationship. The primary key, "*_id*" in table "*Analysts*" links to the foreign key "*analysts*" in table "*Snapshots*". It means that the n (=1, 2, 3, …) analysts in the table "*Snapshots*" depends on the "*_id*" (analyst ID) in the table "*Analysts*". Consequently, the useful data about the system names, people who analyzed them and the analysis date is integrated as the final output shows in Figure 2.1.

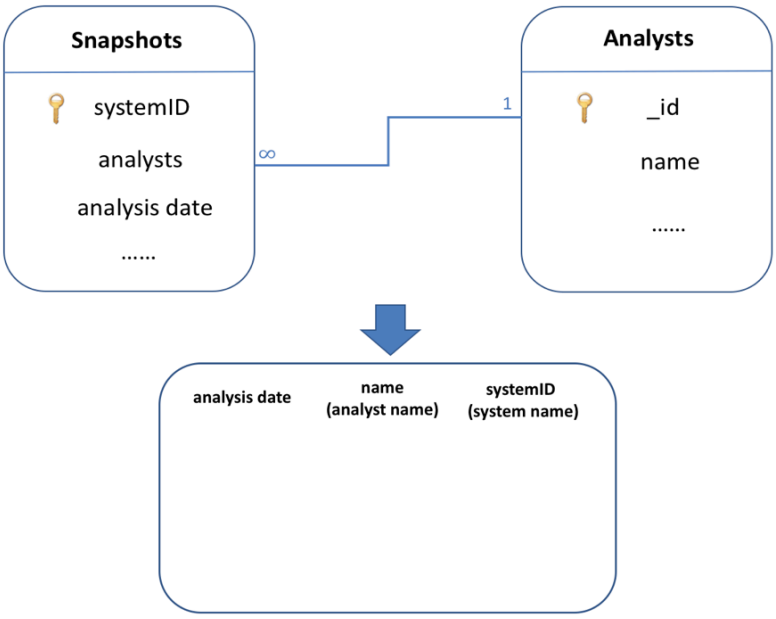


Figure 2.1 Data Extraction

According to this output table, the three columns, "*analysis date*", "*system name*" and "*analyst name*" are needed for our system - industry classification work. Therefore, we are going to efficiently retrieve data from this three-dimensional data set. Data cube, a popular model used to conceptualize the data in the data warehouse, contains points or cells that are measures or values based on a set of dimensions (Datta et al., 1999). Figure 2.2 describes the three dimensions of the data set: "*analysis date*", "*system name*" and "*analyst name*".

Furthermore, several decision support operations are proposed as a part of data analysis process, like slice, dice, drill-down, roll-up and so on. Data modelling offers a lot of solutions for selecting the useful data needed for the classification work. The systems from SAW are within the recent years, from 2008 to 2017. To select these systems in the year order, firstly, we sort the "*analysis date*" from the smallest to the largest and then choose the date that is between 2008 and 2017, it shows as the blue part in Figure 2.3. In Figure 2.4, we reduce the dimensionality of the data by slicing. Slicing refers to selecting the dimensions used to view the cube (Datta et al., 1999). We slice the data for a specific analyst to create a table that consists of the system names and system analysis date. This table can be used for the interview with the specific analysts by providing them the system lists they are familiar with for the classification work.

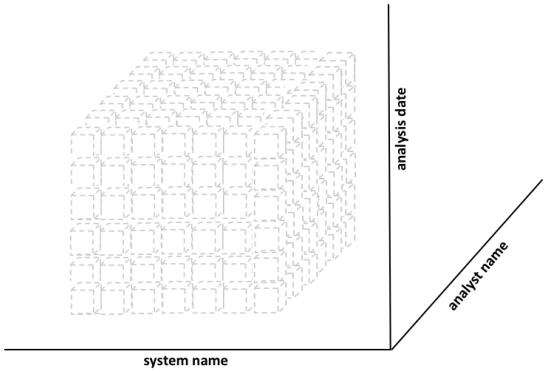


Figure 2.2 Dimensions of Data Set

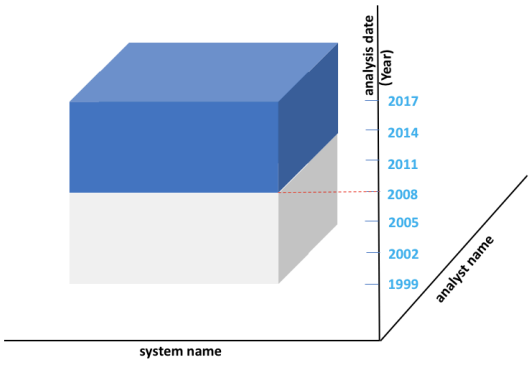


Figure 2.3 Data Selection

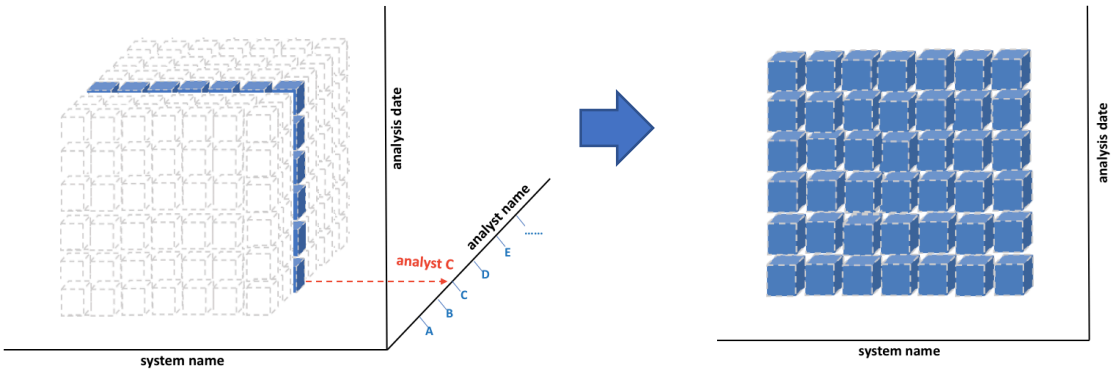


Figure 2.4 Data Slicing

Methodology 2: Interview

Saunders (2011) lists four reasons for using non-standardized (qualitative) research interviews as a method of data collection:

- The purpose of the research. Interviewees may use words or ideas in a particular way, and the opportunity to probe the meanings of these words or ideas will add significance and depth to the data we obtain.
- The significance of establishing personal contact. An interview provides interviewees with an opportunity to reflect on events without needing to write things down. This situation also provides the opportunity for interviewees to receive feedback and personal assurance about the way in which information will be used.
- The nature of the data collection questions. An in-depth or semi-structured interview will be the most advantageous approach to attempt to obtain data in the circumstances where there are a large number of questions to be answered; where the questions are either complex or open-ended; where the order and logic of questioning may need to be varied.
- The length of time required and completeness of the process. Where expectations have been established clearly about the length of time required and participants understand and agree with the objectives of the research interview, they have generally been willing to agree to be interviewed.

Interview with the analysts who are familiar with these systems is needed for our classification work. The interview with the oral explanation on the benchmark makes the categorization work much more efficiently. Therefore, the face-to-face interview will be a better choice than a written or online questionnaire for us.

Methodology 3: Double Checks

Since there are some subjective factors existing in the classification work, it is possible that interviewees will have different opinions, and two interviewees may have different opinions on the classification results. Therefore, the double check which is performed independently can help to improve the accuracy of the final classification results. The methodology "*Double Checks*" is used in our classification work to improve the accuracy of the final categorization results. In our research, the results for each classification are collected by at least two interviewees. And this methodology is used together with Methodology 4: Hierarchical Classification to get the final categories.

Methodology 4: Hierarchical Classification

Classification can be described as the activity of dividing a set of objects into a smaller number of classes in such a way that objects in the same class are similar to one another and dissimilar to objects in other classes (Gordon, 1987). In our case, the benchmark we use is organized in hierarchies. It contains three hierarchical layers, "*Industry*", "*Sector*" and "*Subsector*", which will be described in detail in Section 2.2 later. According to Gauch et al. (1981), most frequently the layers of the dendrogram indicate the average dissimilarity among all sample pairs between the indicated two branches. Figure 2.5 shows a three-layer dendrogram. Each node represents a category, Node H and I are the categories at Layer 0; Node E, F, G are at Layer 1; Node A, B, C, D are the categories at Layer 2. Layer 0 is the layer with the highest hierarchy while Layer 2 has the lowest hierarchy. That is why the category at Node H can be split into the more detailed categories at Node E and F for instance. In our case, finally all the nodes will be joined to the nodes at Layer 0, the highest layer.

The hierarchical classification work is conducted in accordance with the following principles: (Assuming the classification work is done by two interviewees separately.)
• If both Interviewee 1 and Interviewee 2 set the system to Node A, the system is marked as Node A and will be categorized to Node H at last.
• If Interviewee 1 sets the system to Node A while Interviewee 2 sets it to Node B, the system is marked as Node E and will be categorized to Node H at last.
• If Interviewee 1 sets the system to Node A while Interviewee 2 sets it to Node E, the system is marked as Node E and will be categorized to Node H at last.
• If Interviewee 1 sets the system to Node A while Interviewee 2 sets it to Node C or Node F, the system can be categorized to Node H at last as well.
• If Interviewee 1 sets the system to Node A while Interviewee 2 sets it to Node D, or Node G, the categorization work needs to be checked with the third or fourth person, since Node A will be categorized to Node H at last but Node D and G will be categorized to Node I at last. Then the classification of this system will be interviewed with the third or fourth interviewees. If there still exists disagreement for the classification result, the opinions with the majority will be adopted.

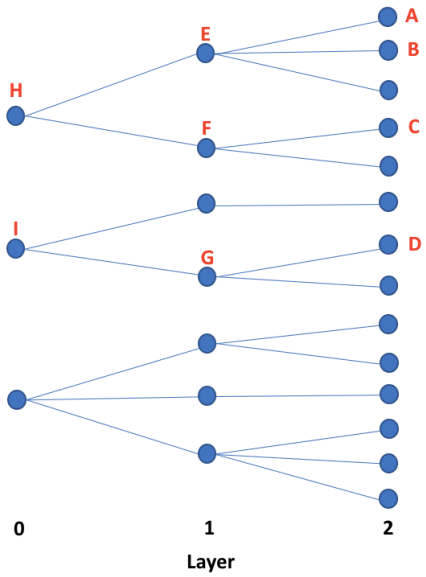


Figure 2.5 Three-layer Dendrogram

2.2 System - Industry Classification

Since the system name is stored with the format: <CUSTOMER> - <SYSTEM> in the data warehouse. <CUSTOMER> is the name of the company, while <SYSTEM> is the name of the system, the system - industry classification work can be transformed to the company - industry classification. First, the systems are categorized into the companies. Then the companies will be classified into the industries they belong to according to their core businesses. From the methodology "*Data Extraction*", the system information including the system names, the analysts who have analyzed them and the analysis date is extracted from the data warehouse. All the systems in SAW are analyzed between 2008 and 2017. Through the "*Data Extraction & Data Modelling*" methodology, 1,519 systems were selected for the interview. The interview will be conducted with 14 analysts in SIG. The system lists as well as the benchmark will be provided to the interviewees.

Compared with the original Industry Classification Benchmark (ICB) retrieved from the official website (http://www.icbenchmark.com/), To make the definition of the industry more concrete, some modifications are made on the benchmark in our industry classification work. The original four-layer benchmark is transformed to the three-layer one, as it is shown in Figure 2.6, by setting the "*Supersector*" layer as the "*Industry*" layer and removing the original "*Industry*" layer. Thus, the "*Industry*" in our benchmark is the same as the "*Supersector*" in the original one. In this way, each "*Industry*" has more concreate definitions compared to the original one. Moreover, according to the guideline 5.2.1 of ICB, A company will be allocated to that "*Subsector*" of ICB whose definition most closely coincides with the source of the company's revenue or the source of the majority of its revenue (FTSE Russell, 2016). It means that ICB just includes the companies with revenue. The government, a non-profit organization, is not included in ICB. However, there are a large number of government systems in SAW. Thus, we create a new industry called Government in the benchmark for the use in our research, as it is shown in Appendix A. Currently, there are 20 nodes at the "*Industry*" layer and each industry has a more detailed definition. For instance, Banking, Financial Services and Insurance supersectors are all related to finical affairs. But in our case, they are considered as different industries based on their different focuses. Like Banking focuses more on the money transmissions, Financial Service mainly provides fiduciary services while Insurance particular deals with insurance related affairs.

During the interview, as it shows in Figure 2.6, the "*Double Checks*" and "*Hierarchical Classification*" methodologies are used. With each interviewee, we are trying to find the specific "*Subsector*" the company belongs to at first. If a company belongs to several subsectors in the same sector, it is categorized into the "*Sector*" layer. Finally, all the companies are classified to the "*Industry*" layer. However, if a company operates the business in multiple subsectors which belongs to different industries, the categorization is based on the dominant business of the company. If the company is finally categorized into different industries from two interviewees' perspectives, it should be marked and checked with other interviewees later. In this case, the opinions with the majority will be adopted.

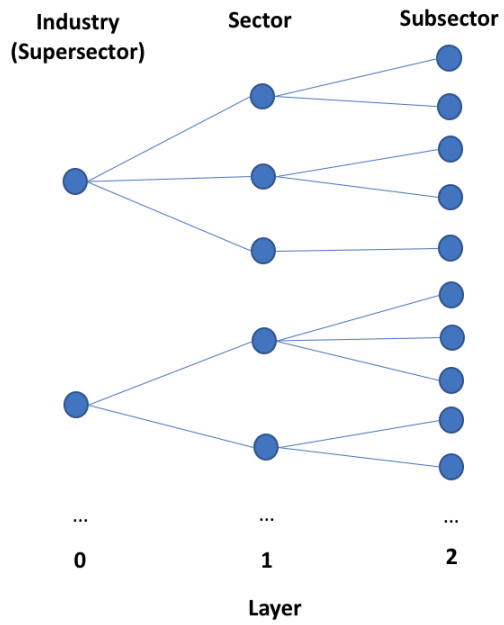


Figure 2.6 Industry Benchmark Dendrogram

2.3 Results

Figure 2.7 describes the system - industry classification results from the interview. 1,519 systems from 172 companies are categorized into 12 industries (supersectors). There is a wide range of system numbers in these industries. Both Banking and Government industries have around 400 systems. However, there are only 3 and 5 systems in the Oil & Gas and the Automobile & Parts industries respectively. Therefore, to minimize the range among the categories, some industries are combined based on their definitions. Oil and Gas supersector which is engaged in the oil and gas exploration is combined with the Utilities supersector which includes the companies focusing on electricity, gas and water generation and distribution. The combination is called the Energy industry. Besides, the newly created industry, Customer Goods industry, consists of Automobile & Parts and Personal & Household Goods. It is worth noting that these combinations of the industries are the adjustments based on our preliminary classification results according to Figure 2.7. The modification on the benchmark within a reasonable scale is allowed.

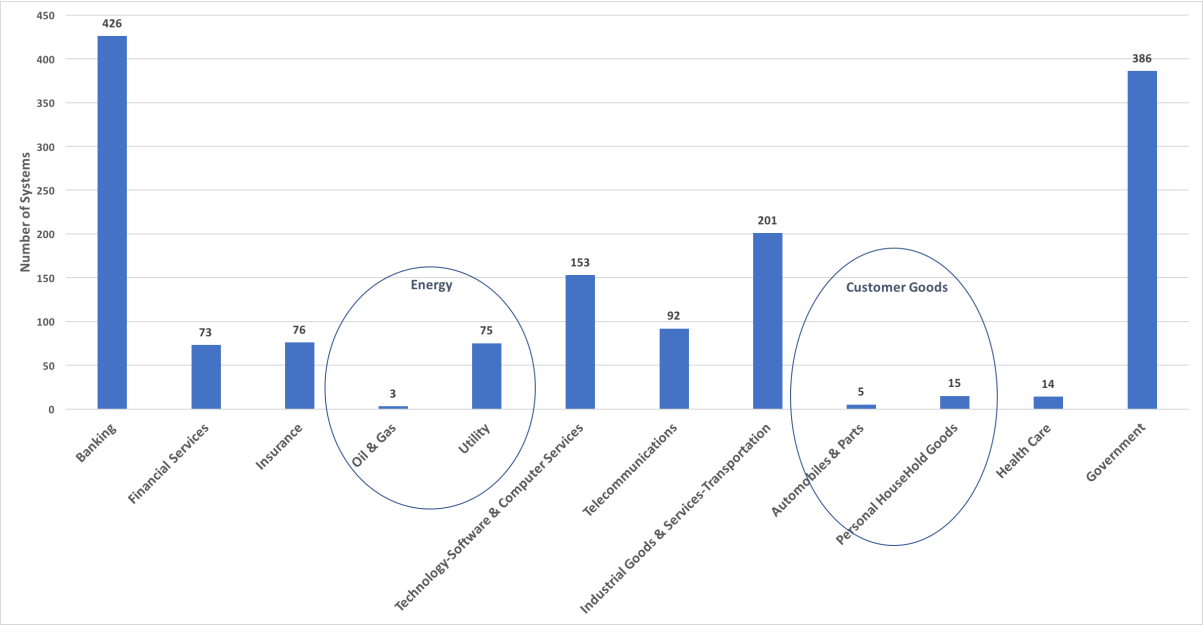


Figure 2.7 System Distribution in Industries (Original)

Consequently, these 1,519 systems are finally categorized into the following 10 industries, as it is shown in Figure 2.8. The definitions of these 10 industries are described as following: (FTSE Russell, 2012).

Banking: Banks provide a board range of financial services, including retail banking, loans and money transmissions.
Financial Services: Companies providing fiduciary services, personal financial services, mortgages, investment services, equity and non-equity investment instruments.
Insurance: Company engaged in life, health, property & casualty and reinsurance.
Energy: The Energy industry is created by combining the Oil & Gas supersector and the Electricity, Gas & Water sectors in the Utilities supersector. It includes the providers and distributers of oil, gas, fuels, water and electricity.
Technology - Software & Company Services: Companies that provide consulting services to other businesses relating to information technology, including providers of computer-system design, system integration, network and system operations, data management and storage repair services and technical support. Or the publishers and distributors of computer software or hardware for home or corporate use.
Telecommunications: Providers of fixed-line telephone services and mobile telephone services.
Industrial Goods & Services - Transportation: Companies providing delivery services, transportation services, marine transportation and railroads.
Customer Goods: It includes the Automobiles & Parts, Food & Beverage and Personal & Household Goods three supersectors.
Health Care: Owners and operators of health maintenance organizations, hospitals, clinic, dentists, opticians, nursing homes, rehabilitation and retirement centers. And the manufacturers and distributors of medical devices and supplies are included as well.
Government: It is not an industry listed in the Industry Classification Benchmark. According to Wikipedia, Government is defined as the public sector concerned with providing various governmental services, like public security, social welfare, urban planning, transportation infrastructure, education and so on.

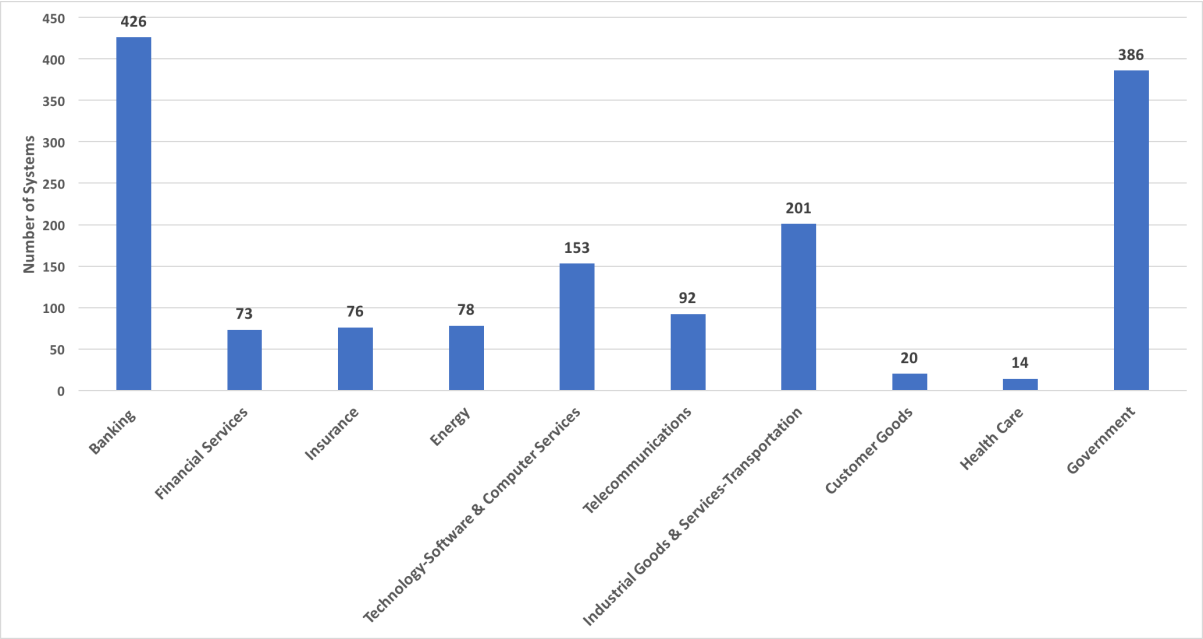


Figure 2.8 System Distribution in Industries (Final)

2.4 Results Discussion

According to the fact-based data classification, the systems in SAW are not equally distributed into10 industry branches. There are much more Banking and Government systems compared with the systems in other industries in SIG's data warehouse. 426 systems are grouped into the Banking industry and 386 systems are grouped into the Government industry. However, Health Care and Customer Goods industries only have 14 and 20 systems respectively. The unequal number of systems in industry branches will influence the results of the comparison among the industries later. Because for some industries we have a large number of data set to be analyzed while for some industries we do not have enough data set to support our final conclusion. This scenario will be illustrated more specifically in Chapter 4 and 5.

Chapter 3 Collecting Commonly Used Technologies from the Systems

To find the commonly used technologies from these 1,519 systems, we are going to make groups of these systems based on their use of technologies and then detect the technologies from these groups. In this chapter, first we describe how we collect the system technologies from SAW and the approaches used to group the systems that have similar technology choices. Besides, the groups created from the data sets, the systems distribution in each group and the technologies that are detected from these groups are shown at the end of this chapter.

3.1 Data Collection

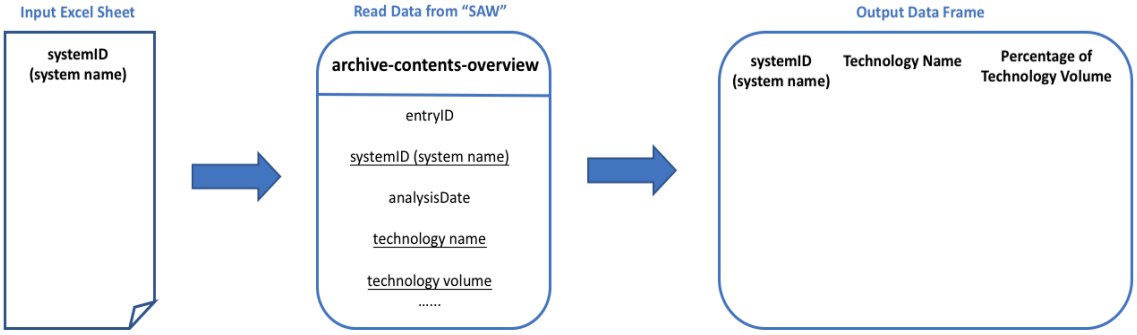


Figure 3.1 System Technology Data Collection

The process of getting the technology data we need for our research is described in Figure 3.1. 1,519 systems are classified into industry branches in the previous chapter. In this chapter, the technology information for those systems is collected.

The Lines of Code (LOC) is usually used to measure the volume of code. However, it can only be used to compare or estimate the projects that use the same language, and are coded using the same coding standards. LOC highly depends on programmers' programming styles. And different time will be taken to write code in different languages for the same LOC. SIG expresses volume as rebuild value in man years. The rebuild value of a system describes how long it would take to rebuild the system based on market average productivity. As it is shown in Figure 3.2, the rebuild value is calculated by multiplying the volume of lines of code with the market average productivity in the technology used and by using the rebuild value, the volume of systems in different technologies can be compared. Furthermore, to remove the impact of the system size on the comparison among different systems, we use the volume percentage ("*Technology Volume*" divided by "*All Aggregate Technology Volume*") to represent the volume usage of each technology for each system.

The *Output Data Frame* has three columns: "*System Name*", "*Technology Name*" and "*Percentage of Technology Volume*" as it is shown in Figure 3.1. Here the "*Technology Name*" appears in the single form, which means if a system uses multiple technologies, the "*System Name*" will be found in multiple lines together with each technology and its volume percentage, as it shows on the left side of Figure 3.3.

Figure 3.2 Methods for Technology Volume Comparison

3.2 Data Transformation

System-Technology Data Frame

System Name	Technology Name	Percentage of Technology Volume
System 1	Java	80%
System 1	JSP	15%
System 1	JavaScript	5%
System 2	Java	90%
System 2	XML	10%
System 3	Java	95%
System 3	XML	5%
System 4	COBOL	100%
System 5	C#	80%
System 5	ASPX	20%
System 6	Java	45%
System 6	JSF	25%
System 6	PL/SQL	15%
System 6	XSD	5%
System 6	.....	.....
.....	.....	.....

System-Technology Combination Data Frame

System Name	Technology Combination List	Percentage of Technology Volume Combination List
System 1	(Java, JSP, JavaScript)	(80%, 15%, 5%)
System 2	(Java, XML)	(90%, 10%)
System 3	(Java, XML)	(95%, 5%)
System 4	(COBOL)	(100%)
System 5	(C#, ASPX)	(80%, 20%)
System 6	(Java, JSF, PL/SQL, XSD, ...)	(45%, 25%, 15%, 5%, ...)
.....	.....	.....
.....	.....	.....

Figure 3.3 Data Frame Transformation

We centralize the technology information for each system by transforming the *System-Technology Data Frame* to the *System-Technology Combination Data Frame*, according to the right side of Figure 3.3. In the new Data Frame, each system's technology information is displayed in one row. There are 1,519 rows and 3 columns in total. The first column shows the system names. The second column lists all the technologies used by each system. Some systems are composed of just one or two technologies, some are composed of three or more technologies, like System 1 and 6 in Figure 3.3. Each technology combination forms a vector, the vectors have different lengths depend on the number of technologies used by each system. And the third column describes the corresponding volume percentage of the technologies in Column 2. Therefore, for each system, the length of the vector in Column 3 is the same as the length in Column 2. The sum of the figures in each vector on the third column is equal to 100%.

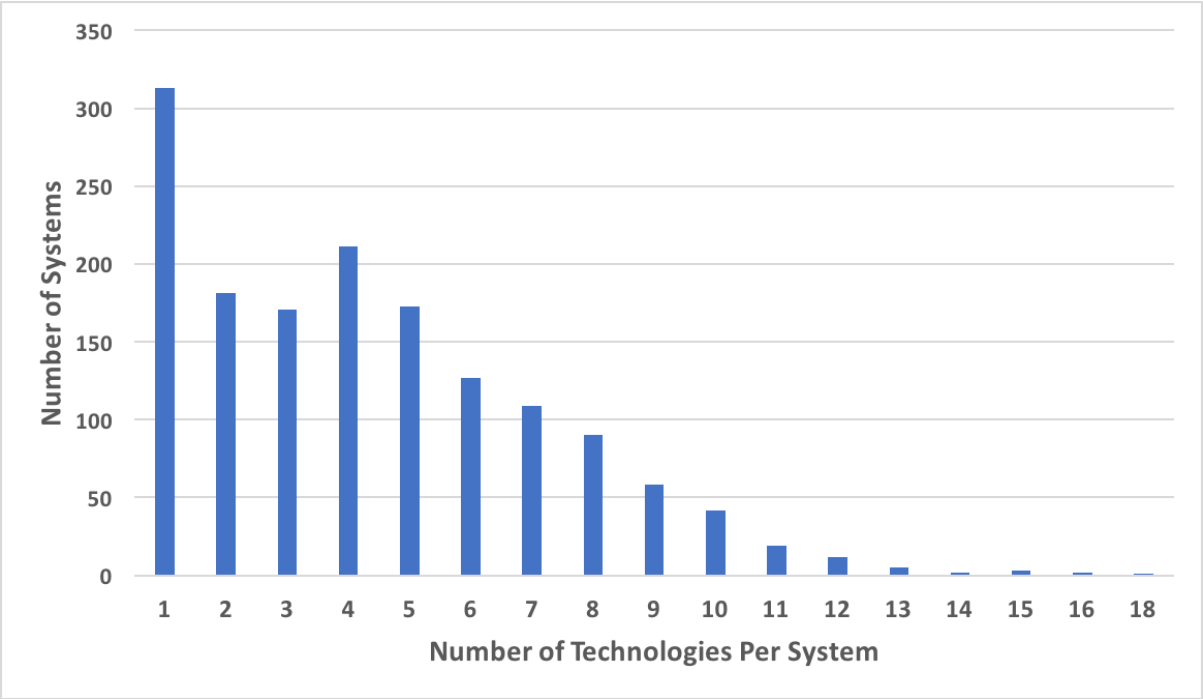


Figure 3.4 System Distribution Depends on the Number of Technologies Per System

Figure 3.4 shows the systems distribution depends on the number of technologies per system. Around 300 systems use only one technology, while there are around 80% systems using more than one technology. Even one system was implemented by using 18 technologies. For large software projects, it is very common to find a mixture of languages used in a system. Usually, this is because software code to be reused (existing system libraries, organizational reuse libraries, or COTS software) is in a language other than the primary language, or else a particular language is required to accomplish a particular function for some special reason. Several languages are interfaced and the language mix will probably produce more reliable results (Lawlis, 1997). From Figure 3.4 we can also see that besides the systems that use only a single technology, most are detected containing two to five technologies. Mixing language is never quite as straightforward as using just one language. The use of two or more development languages together is often more trouble than it is worth (Lawlis, 1997). Thus, as it is shown in Figure 3.4, the "*Number of Systems*" goes down generally with the "*Number of Technologies Per System*" goes up.

3.3 System Grouping Model

We are going to group the systems based on their use of technology. It can be regarded as a clustering process to group the unlabeled data. According to Jain et al. (1999), Clustering is the unsupervised classification of *patterns* (observations, data items, or feature vectors) into groups (clusters). The objective of clustering is to partition a set of unlabeled objects into homogeneous groups or clusters (Fred et al., 2005). But the category labels are data driven; that is, they are obtained solely from data (Jain et al., 1999). Therefore, during the clustering process, we are going to create the category labels, which are called "*technology stacks*" in our research. Refer to Hunt et al. (2007), The software stack is formed by the operating systems which embody a collection of design decisions. Similarly, the *technology stack* contains the technology decisions about the main technology combinations or the single technology that are relatively commonly used while implementing the systems. And then based on the labels, the systems are grouped into clusters. Each *technology stack* can be regarded as one cluster. The

systems in the same cluster have higher similarity of the technology choice compared with the systems in other clusters.

Jain et al. (1999) describe the "*Stages in Clustering*" as Figure 3.5 depicts. "*Feature selection* is the process of identifying the most effective subset of the original *features* to use in clustering. *Feature extraction* is the use of one or more transformations of the input *features* to produce new salient *features*. Either or both of these techniques can be used to obtain an appropriate set of *features* to use in clustering. *Pattern representation* refers to the number of available *patterns*, and the number, type, and scale of the *features* available to the clustering algorithm. After measuring the similarity of the *patterns*, they are grouped into clusters. The *patterns* within a valid cluster are more similar to each other than they are to a pattern belonging to a different cluster. Additionally, the grouping process output could affect subsequent *feature extraction* and similarity computations". The Grouping step can be performed in a number of ways. Traditional clustering techniques can be broadly classified into two categories: partitional and hierarchical. Partitional clustering obtains a partition of the objects into clusters such that the objects in a cluster are more similar to each other than to objects in different clusters; A hierarchical clustering is a nested sequence of partitions. It starts by placing each object in its own cluster and then merges these atomic clusters into larger and larger clusters until all objects are in a single cluster (Agrawal et al., 2005). Both these two clustering techniques are based on similarity measurement, which is always carried out by measuring the distances among the objects.

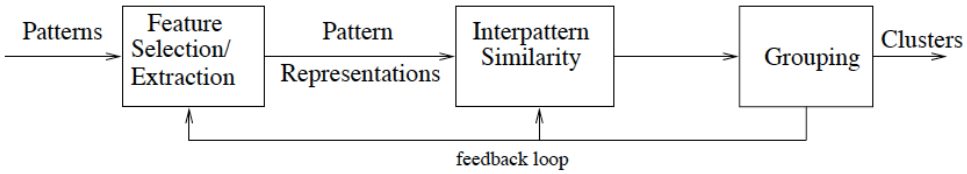


Figure 3.5 Stages in Clustering

In our research, the *feature* is represented by the technologies used for the system implementation. However, according to Figure 3.4, the number of technologies used for implementing systems varies from 1 to 18. It is hard to define a certain number of technologies that should be extracted from all the systems to represent the *feature* of the systems. Thus, for each system, the number of technologies selected for grouping should be better based on the actual number of technologies used by it. For example, if a system is implemented using Java, JavaScript and HTML, selecting all these three technologies to represent the *features* of the system is more accurate than only selecting one technology. Furthermore, the data matrix in Table 3.1 performs the data set in Figure 3.3 in another form. The matrix contains 1,519 rows and 153 columns. The rows represent the data objects, that are the system names, and the columns are the attributes of the data set objects, the technologies used in each system. It means that there are 153 unique technologies altogether in these 1,519 systems. The elements in the matrix show the percentage occupation of each technology among the total volume of technology for a certain system. All the technologies even with only 1% percentage are included in the matrix and the figures are rounded to the integer, which means 79.5% ≈ 80.4% ≈ 80% for example. Thus, the sum of the numerical values for each row equals to 1. If the system does not use the technology, the percentage is set to 0. Each column is regarded as a dimension of these 1,519 objects, and there are 153 dimensions in the data matrix. It is indeed a high-dimensional data set. As the number of dimensions in a data set increases, distance measures become increasingly meaningless (Parsons et al., 2004). Therefore, measuring the

distances to group the systems is not in the scope of our research according to the nature of our data set.

Table 3.1 System - Technology Data Matrix

	Java	JSP	JavaScript	XML	COBOL	C#	ASPX	JSF	PL/SQL	XSD	.....
System 1	80%	15%	5%	0	0	0	0	0	0	0	0
System 2	90%	0	0	10%	0	0	0	0	0	0	0
System 3	95%	0	0	5%	0	0	0	0	0	0	0
System 4	0	0	0	0	100%	0	0	0	0	0	0
System 5	0	0	0	0	0	80%	20%	0	0	0	0
System 6	45%	0	0	0	0	0	0	25%	15%	5%	.....
.....											

Considering the problems described in the previous paragraph, the "*Stages in Clustering*" model described in Figure 3.5 is not fully applicable to our case. Therefore, a new model, "*Stages in System Clustering*" is created for our grouping work, according to Figure 3.6. *Patterns* are the 1,519 systems as well as their *features*, the technologies. These are included in the *System-Technology Combination Data Frame*, as it is shown on right side of Figure 3.3. The technology information, including the name and the volume percentage of the technologies used to implement the systems, are the *features* of the systems in our research. According to Bouwers (2013), "The factor *information extent* checks the amount of information needed to understand the implemented architecture. The more technologies are used, the bigger the total extent of information will be". For instance, if a system is implemented using Java, JavaScript and HTML, the *information extent* provided for this system with only Java language is smaller than the *information extent* provided with all these three languages. Undoubtedly, for the systems that use multiple technologies, the larger *information extent* selected will be the better *features* to identify these *patterns*. Thus, during the *Feature Selection/ Extraction* stage, based on the volume percentage, the dominant technologies are selected from each system. And for each of them, a technology combination vector which contains these technologies is generated. The systems with the dominant technologies are the *Pattern Representations* prepared for the *Grouping* phase. The systems with the same technology combination vector are gathered in the *Grouping* step. How the technologies are selected for each system, how the new technology combination vectors are formed and how the systems are grouped by the same vectors will be explained in detail in Section 3.4 later. Besides, the threshold has to be set to select the *patterns* to be the clusters, which means if the number of data items exceeds the threshold, these *patterns* are selected to form a cluster. The rest *patterns*, all except the selected ones, go for the next loop. During the next loop, the *features* of the *patterns* will be reselected with 1 less technology collected from each system. The loop will stop until the number of technology collected from systems becomes 0.

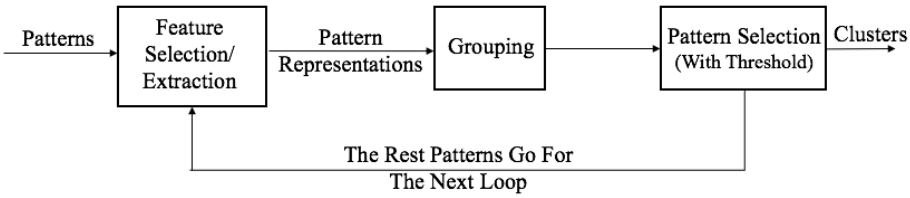


Figure 3.6 Stages in System Clustering

3.4 System Grouping Algorithm

According to Knuth, an algorithm has five important features (Knuth, 1973):
1. **Fitness**: The algorithm must always terminate after a finite number of steps.
2. **Definiteness**: Each step must be precisely defined.
3. **Input**: An algorithm has zero or more inputs, taken from a specified set of objects.
4. **Output**: An algorithm has one or more outputs, which have a specified relation to the inputs.
5. **Effectiveness**: All operations to be performed must be sufficiently basic that they can be done exactly and in finite length to achieve the goal.

Besides, an algorithm can be expressed in a number of ways, including natural languages, flow charts, pseudo-code and programming languages. The flowchart in Figure 3.7 describes the algorithm based on the model created in Figure 3.6 for the system grouping. In this algorithm, three Inputs are needed:
- The assumption of the "*Lower Limit Number of Systems Per Stack*";
- The initial length of *technology stack*, which is set to 18. Because according to Figure 3.4, the largest number of technology used by the system is 18, which means, there is no *technology stack* created that can be longer than 18 in our research.
- The *System-Technology Combination Data Frame* (refer to the right part of Figure 3.3).

During this process, starting from the initial length of the stack 18 to 1, the program runs for 18 iterations. It means that in the first iteration, the length of the stack is defined as 18. Similarly, the length is defined as 17 in the second iteration, then 16, 15… In each iteration, the number of technologies collected from the systems is equal to the defined length of the stack. Thus, at first, only the systems using at least that number of technologies are chosen and their *features* (dominant technologies) are selected and extracted directly from the original *System-Technology Combination Data Frame* to generate the new technology combination vectors with the certain length. For example, (Java, XML) is a technology combination vector with the length of 2 and (Java) is a vector with length 1. Then the systems that have the same technology combination vectors are grouped together. The vectors that contain the same technologies but with different orders of the technologies are regarded as the same vector. For instance, (Java, XML) and (XML, Java) are the same. Based on the input assumption value, the "*Lower Limit Number of Systems Per Stack*", the eligible technology combination vectors and the systems that match to these vectors are selected. These technology combination vectors are viewed as the *technology stacks* then. Besides, these systems as well as the *technology stacks* are added to the new data frame, the *System-Technology Stack Data Frame*. Meanwhile, these systems are ruled out from the original data frame for the next loop. Since the length of the stack is defined as a certain number in each iteration and the stack with larger length is created prior to the one with shorter length, the systems which are already grouped into the stacks with larger length will not participate in the next grouping iteration. In this way, the algorithm makes sure that each system is grouped into only one *technology stack*. After all the loops, the new Data

Frame that contains the *technology stacks* with different lengths and the corresponding systems' information is output.

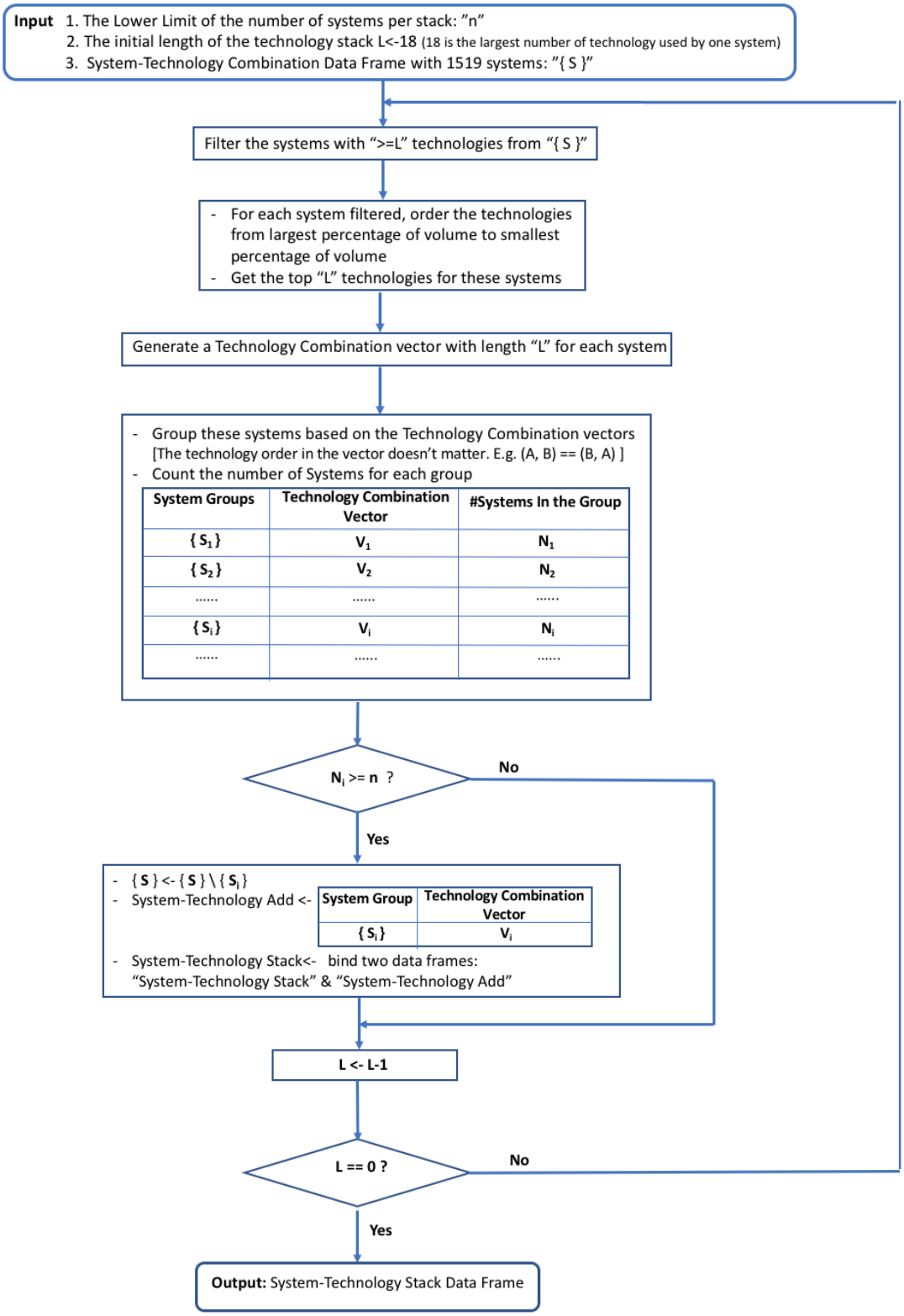


Figure 3.7 Systems Grouping Algorithm

However, according to the algorithm described in Figure 3.7, the input value, the "*Lower Limit Number of Systems Per Stack*" should be assumed at the beginning. With different input value,

we get different number of stacks as well as different total number of systems that can be clustered from this algorithm. Figure 3.8 and Figure 3.9 have the same horizontal axis but different vertical axes. The relation between the "*Number of stacks*" and the "*Lower Limit Number of Systems Per Stack*" is depicted in Figure 3.8. It is obvious that the "*Number of stacks*" decreases rapidly with the growth of the "*Lower Limit Number of Systems Per Stack*" from 1 to 10, while it declines much more slowly when the lower limit is larger than 10. And Figure 3.9 illustrates the relation between the "*Total Number of systems Being Grouped*" into stacks and the "*Lower Limit Number of Systems Per Stack*". As it can be seen from the graph, when the lower limit is set to 50 or so, only 66% systems are grouped, which means around 1/3 systems are ungrouped under this situation. Both Figure 3.8 and Figure 3.9 demonstrate the variation of the output based on different assumed input values. For instance, if the lower limit is set to 1, which means one system can be a cluster, all the 1,519 systems (100%) will be grouped into 849 clusters. In this situation, there are too many clusters, but only a few systems in each cluster. It is unreasonable to say the *technology stack* is commonly used. For another example, if the lower limit is set to 100, which means only if there are no less than 100 systems in the same technology combination vector, the technology combination vector can be output as a cluster (*technology stack*). However, with the threshold as 100, we can only get 3 stacks with 59% systems being grouped. It means many systems are treated as the outliers in this situation. It is also unreasonable to leave nearly 41% systems ungrouped. According to Sarstedt et al. (2014), It is crucial to ensure that the results are interpretable and meaningful. Not only must be the number of clusters small enough to ensure manageability, but each segment should also be large enough to warrant strategic attention. Therefore, we come to the question that how to set the threshold for the "*Lower Limit Number of Systems Per Stack*" while creating the *technology stack*? In other words, the question is: For considering a technology combination vector common, at least how many systems should be grouped into this vector?

Similar to many cluster analyses, the number of clusters is unknown. However, the correct number of clusters of different types of data sets is seldom known in practice. To identify the number of clusters is an important task and must be faced with many operational challenges. Sometimes it needs the expert domain knowledge over the underlying data sets (Kishor, 2014). Accordingly, we should revert to practical considerations. In our research, we are aiming to group as many systems as possible into clusters. Meanwhile, we need to expand the *information extent* of each cluster (*technology stack*). According to the findings from Figure 3.8 and 3.9, the smaller the figure of the "*Lower Limit Number of Systems Per Stack*", the larger the "*Percentage of Systems Being Grouped*" and the more detailed technology information is provided by each stack. On the contrary, the smaller the figure of the "*Lower Limit Number of Systems Per Stack*" leads to larger number of stacks with very few systems in each stack. Thus, we are going to reduce the number of clusters (*technology stacks*) in a condition that ensured the larger total number of systems being grouped as well as the bigger *information extent* provided by each *technology stack*. It is said by Kodinariya et al. (2013), "By rule of thumb, the approach to select the right number of clusters which can be applied to any type of data set is $K \approx \sqrt{(\frac{n}{2})}$, where n is the number of data points." It is drawn from the experiments that this approach ensures that each segment is large enough to warrant strategic attention to a large extent. Since there are no other suitable and reasonable ways to define the K value according to our situation, we will try this formula: $K \approx \sqrt{(\frac{n}{2})}$. Consequently, in our research, $K \approx \sqrt{(\frac{1519}{2})} \approx 28$. From the partial enlarged diagram inside Figure 3.9, we can see that when K, the number of stacks, is equal to 28, then the "*Lower Limit Number of Systems Per Stack*" is set to 23. And this time, around 78% systems are grouped into 28 clusters. It means that if there are no less than 23 systems in the group that share the same technology combination vector,

we will say that the combination of used technologies is common. There are 28 *technology stacks* created through this method. Moreover, it also ensures that a relatively large number of systems with 78% are grouped.

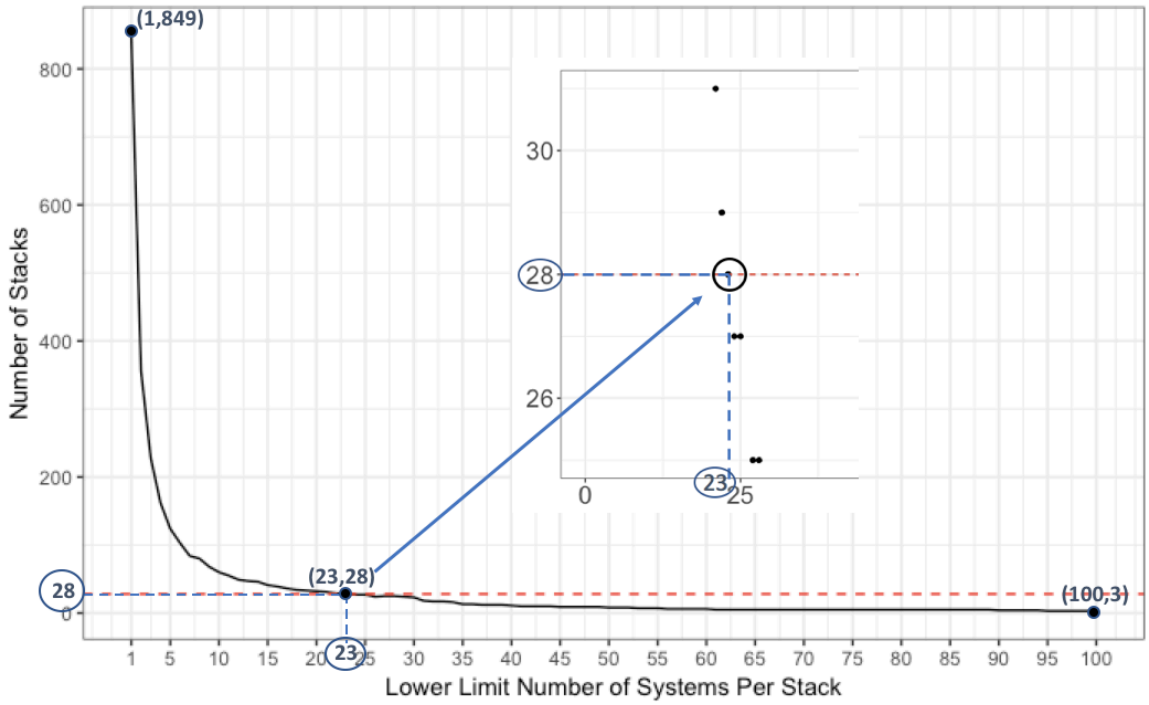


Figure 3.8

Relation Between the Number of Stacks & the Lower Limit Number of Systems Per Stack

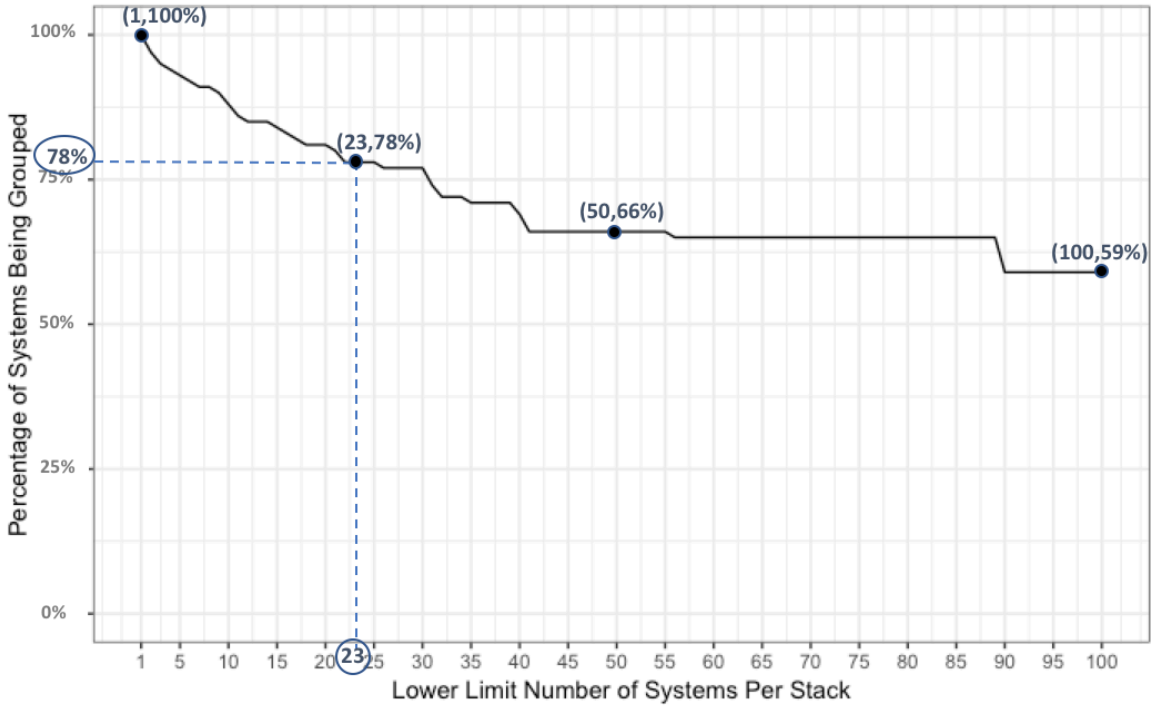


Figure 3.9

Relation Between the Percentage of Systems Being Grouped & the Lower Limit Number of Systems Per Stack

However, if we set different thresholds for the "*Lower Limit Number of Systems Per Stack*", we will get different number of *technology stacks* and the total number of technologies that can be collected from the stacks will be different as well. If the threshold is lower than 23, we will get more than 28 *technology stacks*, and there will probably be more technologies altogether extracted from the *technology stacks*. While if the threshold is higher than 23, less *technology stacks* as well as the total number of technologies will be extracted. The technology extraction from the *technology stacks* will be illustrated in detail in Section 3.5.2.

3.5 Results

As it is shown in Figure 3.10, 78% (1,186 systems) are grouped into certain *technology stacks* while the rest 22% (333 systems) are ungrouped based on the grouping methodology described in the previous section.

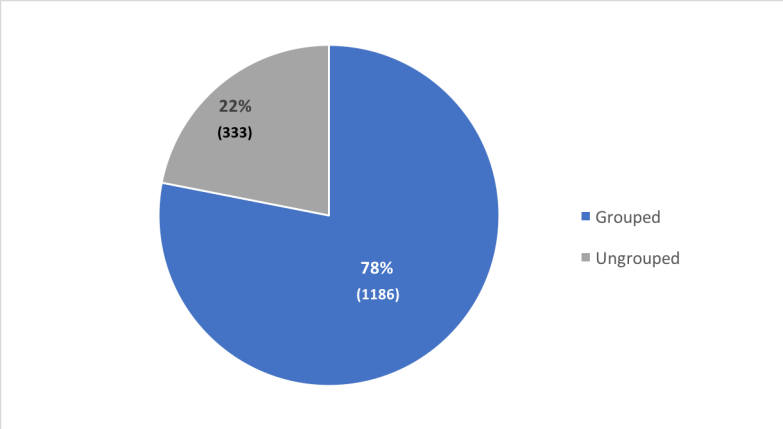


Figure 3.10 Percentage of Grouped and Ungrouped Systems

3.5.1 Systems - Technology Stack Categorization

1,186 out of 1,519 systems are grouped into 28 *technology stacks*. Figure 3.11 demonstrates the number of systems in each *technology stack*. And the *technology stacks* are sorted from the one with the largest number of systems to the one with the smallest number of systems. For instance, the graph shows that most systems are grouped into the (Java) stack. The (Java) stack contains the systems that were only implemented using Java and the systems whose most dominant technology is Java. Similarly, the (Java, XML) stack contains the systems that were only implemented using Java and XML as well as the systems with Java and XML as the two most dominant technologies. It doesn't matter Java and XML which technology occupies the largest percentage of the volume and which one occupies the second largest percentage. Additionally, since the approach we used makes sure the stack with larger length is created prior to the one with shorter length. It means that the (Java, XML) stack is created one iteration before the (Java) stack is being created. Thus, the systems that pertain to the (Java, XML) stack are not included in the (Java) stack. Based on our approach, each system is grouped into only one stack. (Java) and (Java, XML) are two different stacks.

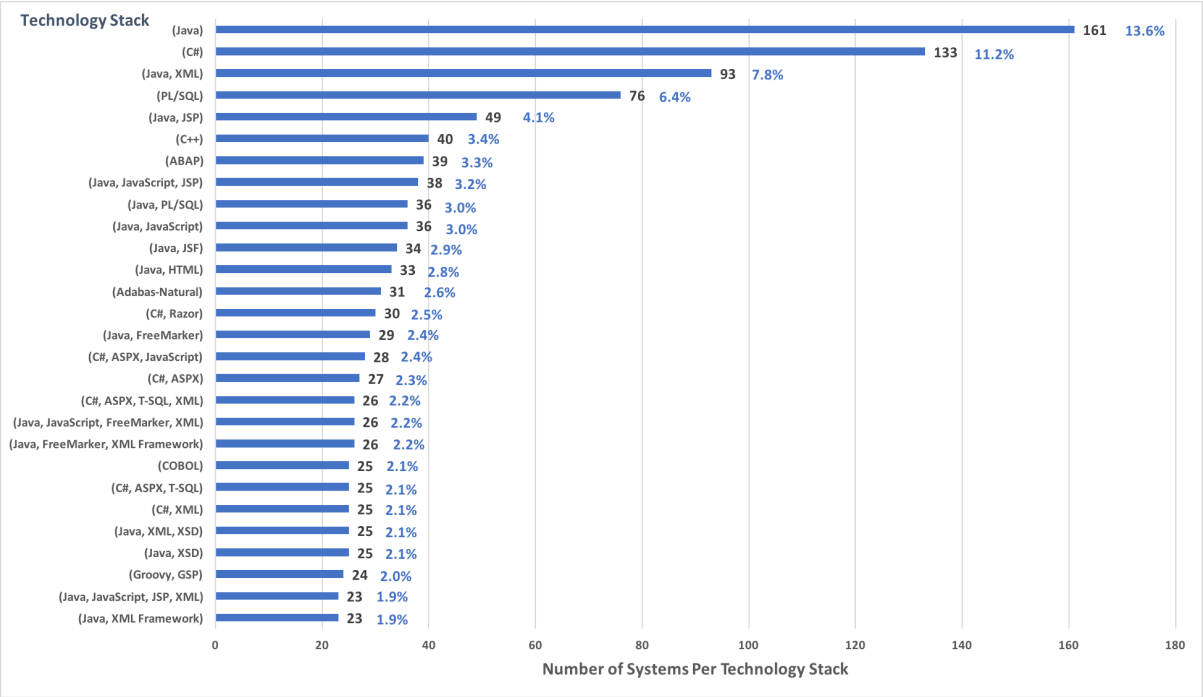


Figure 3.11 System Distribution in Technology Stacks

Among these 28 stacks, some contain only one technology, while some contain two, three or four technologies, which means some technologies can be used alone as the dominant technology for the system implementation, while some technologies are always used together with others. For example, it is rarely to see ASP.NET (ASPX) works alone as the dominant technology. It always works together with C#, while JSF, JSP are always used together with Java. It depends on the technology merits as well as the functionality provided by the technologies for the system implementation. Some technologies can take place of the others, like both JSP and JSF are able to generate web pages. Besides, some technology combinations are alternative to the others, like (Java, JSF) and (C#, ASPX) are able to provide similar functionality. The technology comparison should be better based on the technologies that work on the same purpose. Therefore, we are going to categorize the technologies that appear in these 28 *technology stacks* into different groups according to the functionality types of these technologies.

3.5.2 Technology – Functionality Type Categorization

The 20 technologies that appear in these *technology stacks* are listed in Table 3.2, which are alphabetically sorted. The *technology stacks* which contain these technologies and the number of *technology stacks* are described in the table as well. Moreover, since XML and XML Framework are the same things, we combine them together as XML in the analysis. Therefore, we have 19 technologies in total. It means that according to our data set, these 19 technologies are widely used as the dominant technologies for the system implementation.

Table 3.2 Technology with Its Related Technology Stacks

Technology	Related Technology Stacks	Number of Technology Stacks
ABAP	(ABAP)	1
Adabas-Natural	(Adabas-Natural)	1
ASPX	(C#, ASPX) (C#, ASPX, JavaScript) (C#, ASPX, T-SQL) (C#, ASPX, T-SQL, XML)	4
C#	(C#) (C#, Razor) (C#, XML) (C#, ASPX) (C#, ASPX, T-SQL) (C#, ASPX, T-SQL, XML)	6
C++	(C++)	1
COBOL	(COBOL)	1
FreeMarker	(Java, FreeMarker) (Java, FreeMarker, XML Framework) (Java, JavaScript, FreeMarker, XML)	3
Groovy	(Groovy, GSP)	1
GSP	(Groovy, GSP)	1
HTML	(Java, HTML)	1
Java	(Java) (Java, JSP) (Java, JavaScript) (Java, JSF) (Java, HTML) (Java, FreeMarker) (Java, XML) (Java, XSD) (Java, XML, XSD) (Java, XML Framework) (Java, PL/SQL) (Java, JavaScript, JSP, XML) (Java, JavaScript, JSP) (Java, FreeMarker, XML Framework) (Java, JavaScript, FreeMarker, XML)	15
JavaScript	(Java, JavaScript) (Java, JavaScript, JSP) (Java, JavaScript, JSP, XML) (Java, JavaScript, FreeMarker, XML)	4
JSF	(Java, JSF)	1
JSP	(Java, JSP) (Java, JavaScript, JSP) (Java, JavaScript, JSP, XML)	3
PL/SQL	(PL/ SQL) (Java, PL/SQL)	2
Razor	(C#, Razor)	1
T-SQL	(C#, ASPX, T-SQL) (C#, ASPX, T-SQL, XML)	2
XML (XML & XML Framework)	(Java, XML) (C#, XML) (Java, XML, XSD) (Java, JavaScript, JSP, XML) (Java, JavaScript, FreeMarker, XML) (C#, ASPX, T-SQL, XML) (Java, XML Framework) (Java, FreeMarker, XML Framework)	8
XSD	(Java, XSD) (Java, XML, XSD)	2

Furthermore, the definition of each technology, the functionality it is able to provide to support the systems and the *technology stacks* that are related to that technology are listed in Appendix B. Owing to the confidentiality of the source code, it is hard to figure out the exact functionality of the technologies while implementing the systems. Therefore, the functionality categories made for these 19 technologies are only based on the literature review. We assume that the technologies are providing similar functionality as they do generally for the system implementation. The definitions and the functionality descriptions of these technologies are

obtained from non-scientific literature, which is called "*Grey Literature*" as well. It includes the literature that is obtained from Google, Wikipedia and blogs, or some online communities, like Stack Overflow and Quora. This information is collected from experienced software engineers and programmers. It adds values to the work of identifying the functionality type of these 19 technologies.

According to the "*Functionality Description*" column in Appendix B, some technologies are able to provide multiple functionalities while some are focusing on a specific application domain, like the technologies that only target on web applications, database or data exchange. Additionally, some technologies appear in the stack alone, some are always detected in the stack together with a certain technology. As a result, these 19 technologies are categorized into different types, according to their functionalities as well as the related *technology stacks*. Figure 3.12 provides an overview of the *Technology Functionality Type* and Table 3.3 shows the final categorization of these technologies.

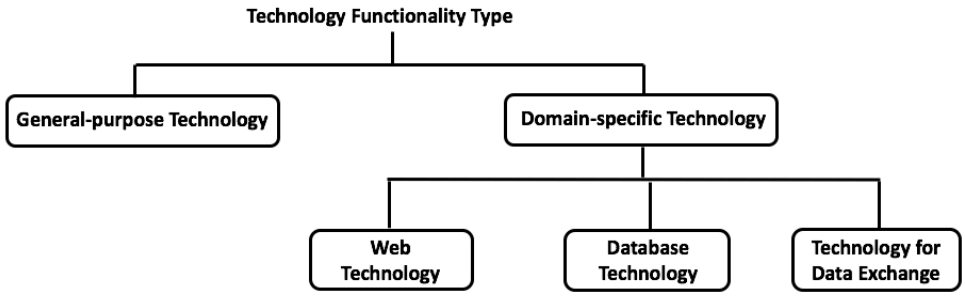


Figure 3.12 Technology Functionality Type

The definitions of these types are listed as following:

- **General-purpose Technology**. The technologies that can be used for writing software in a wide variety of application domains. Most of them are able to provide multiple functionalities. And these technologies can work alone as the dominant technologies for the system implementation including building web applications, connecting database, processing data, generating reports and so on.

- **Domain-specific Technology.** The technologies that have specialized features for a particular domain. Since they are only focusing on a specific domain, most of them always work together with other technologies for the system implementation.

    - **Web Technology**. The front-end technologies. They are used for creating web pages and web applications.

    - **Database Technology**. The technologies that are used for interacting with the Database Management System.

    - **Technology for Data Exchange**. The technologies that are used for managing data. They make it much easier to create data that can be shared by different applications.

Table 3.3 Technology List in Functionality Type

Technology Functionality Type	General-purpose	Domain-specific		
		Web	Database	Data Exchange
Technology List	ABAP Adabas-Natural C# C++ COBOL Groovy Java	ASP.NET(ASPX) FreeMarker GSP HTML JavaScript JSF JSP Razor	PL/SQL T-SQL	XML XSD

According to Appendix B, ABAP, C#, C++, COBOL and Java are able to provide multiple functionalities, including building web applications, connecting database, processing data, generating reports and so on. Thus, they are categorized into the General-purpose type. As for Groovy, which is useful as both a scripting language and also as a general-purpose language, is regarded as a general-purpose technology in our research. As for Adabas-Natural, it works for database systems majorly. However, according to our data set, it always works alone as the only system implementation technology. Thus, it is categorized into General-purpose category in our research as well. Among these 28 *technology stacks*, each stack contains one general-purpose technology and there is only one general-purpose technology appears in the stack. Besides, the other technologies with specialized features for a particular domain are rarely detected being used alone. They are always used together with a general-purpose technology. These technologies are categorized into the type of Domain-specific. ASP.NET, FreeMarker, GSP, HTML, JavaScript, JSF, JSP and Razor are web technologies; PL/SQL and T-SQL are database technologies; XML and XSD are the technologies for data exchange. The results are concluded in Table 3.3.

3.5.3 Abstract Stack

Since the 19 technologies that appear in the *technology stacks* are categorized into corresponding functionality types. Based on it, the 28 *technology stacks* can be categorized into the groups of *Technology Functionality Type Combination*. The group of *Technology Functionality Type Combination* is called "*abstract stack*" in our research. For example, Java is a general-purpose technology, JavaScript is a web technology, so the *technology stack* (Java, JavaScript) is grouped into the *abstract stack* (General-purpose, Web) group, and so as the *technology stack* (Java, JSP) and (C#, ASPX). Table 3.4 lists the 8 *abstract stacks* and the corresponding *technology stacks* that belong to each category. The *technology stacks* that are in the same *abstract stack* are assumed to be able to provide similar combination of the functionality for the system implementation.

Table 3.4 Categorizing Technology Stacks into Abstract Stacks

Abstract Stack (Technology Functionality Type Combination)	Technology Stack
(General-purpose)	(Java) (C#) (C++) (ABAP) (COBOL) (Adabas-Natural)
(Database)	(PL/SQL)
(General-purpose, Web)	(Java, JSP) (Java, JavaScript) (Java, JavaScript, JSP) (Java, JSF) (Java, HTML) (Java, FreeMarker) (C#, Razor) (C#, ASPX) (C#, ASPX, JavaScript) (Groovy, GSP)
(General-purpose, Database)	(Java, PL/SQL)
(General-purpose, Data Exchange)	(Java, XML) (Java, XSD) (Java, XML, XSD) (Java, XML Framework) (C#, XML)
(General-purpose, Web, Database)	(C#, ASPX, T-SQL)
(General-purpose, Web, Data Exchange)	(Java, JavaScript, FreeMarker, XML) (Java, FreeMarker, XML Framework) (Java, JavaScript, JSP, XML)
(General-purpose, Web, Database, Data Exchange)	(C#, ASPX, T-SQL, XML)

As it is shown in Table 3.4, almost all the *abstract stacks* contain the general-purpose technology. General-purpose technology can be used alone as the dominant technology for the system implementation. That is why there is an *abstract stack* called (General-purpose). While web technologies, database technologies and the technologies for data exchange are always used together with a general-purpose technology.

Furthermore, the web technologies: FreeMarker, JSF, JSP, JavaScript and HTML are always used together with Java; ASP.NET and Razor are used together with C#; and GSP is together with Groovy. Besides, the database technology PL/SQL is frequently used along with Java, while T-SQL is found with C#. As for the technologies for data exchange, XML and XSD are found used with both Java and C#.

According to the "*Definition*" column in Appendix B, the reasons for these findings could be: FreeMarker, JSF, JSP are the Java-based web technologies, they are always used for building user interfaces and web applications for Java programs. Moreover, Java was originally developed by Sun Microsoft but now owned by Oracle Corporation. And PL/SQL is Oracle Corporation's procedural extension for SQL and the Oracle relational database. Thus, these technologies are always used together. Similarly, C#, ASP.NET and T-SQL are the technologies developed by Microsoft Corporation, and that is why these technology combinations exist more often. As for the other technologies, XML and XSD can be added into the *technology stacks* with Java or C# for the data exchange functionality. As for JavaScript and HTML, they can be used together with both Java and C# theoretically. But according to

our data sets, they are much more frequently existing in the *technology stacks* together with Java for adding values to the web page generation.

3.5.4 Technology Popularity in Each Technology Functionality Type

The systems distribution in each functionality type of the technology is illustrated in Appendix C. Each system is categorized into only one *technology stack* and the *technology stack* contains the dominant technology of that system. In other words, for each technology, the technology can be regarded as the dominant technology for the system, as long as the system is grouped into the *technology stack* that contains this technology. Therefore, the popularity of the technology working as the dominant technology for the system implementation can be obtained by counting the number of systems that are grouped into the *technology stacks*, in which the technology is included.

The data from Appendix C shows that:

- In General-purpose type, based on SIG's data set, there are much more systems that are using Java and C# as their dominant technologies compared with other general-purpose technologies. And between Java and C#, in general, Java is more frequently used than C#. There are more than two times of the systems using Java as their dominant technologies compared with the number of systems that use C#.
- There are 8 popular web technologies according to SIG's data set. The order of the popularity of these 8 technologies are: JavaScript, JSP, ASP.NET, FreeMarker, JSF, HTML, Razor, GSP.
- According to the data set, 112 systems use PL/SQL as their dominant database technologies while 51 use T-SQL.
- XML is the most widely used technology for the data exchange purpose. It is used much more frequently compared with XSD.

3.6 Results Discussion

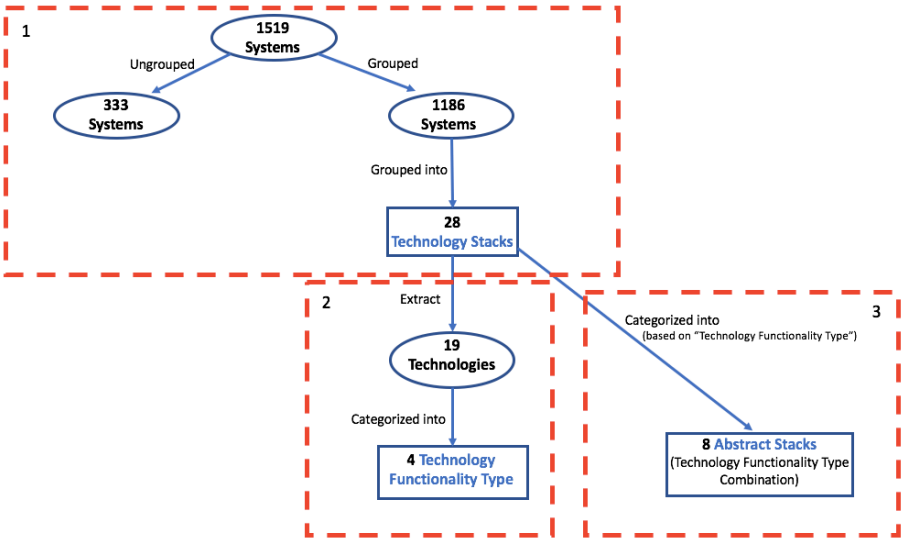


Figure 3.13 System-Technology Grouping Results

Figure 3.13 gives an overview of the procedure about how we get the results for this chapter. We grouped 78% systems (1,186 out of 1,519 systems) into 28 *technology stacks*. Each of the 1,186 systems is grouped into only one *technology stack*. Based on these *technology stacks*, 19

technologies in total are extracted from the stacks, which means these 19 technologies are widely used as the dominant system implementation technologies, according to the data from SIG's data warehouse. Then we categorized these technologies into General-purpose, Web, Database and Data Exchange categories based on the functionality type of these technologies. ABAP, Adabas-Natural, C#, C++, COBOL, Groovy and Java are general-purpose technologies; ASP.NET, FreeMarker, GSP, HTML, JavaScript, JSF, JSP and Razor are web technologies; PL/SQL and T-SQL are database technologies; XML and XSD are the technologies for data exchange. Additionally, based on the *Technology Functionality Type*, the 28 *technology stacks* are grouped into 8 *abstract stacks* (the groups of *Technology Functionality Type Combination*). From the *abstract stacks*, we find that almost all the *abstract stacks* contain the general-purpose technology, and every web technology, database technology or data exchange technology has to be used together with a general-purpose technology. Java and C# are the most frequently used general-purpose technologies. Moreover, Java is always used together with JSP, JSF, FreeMarker, JavaScript, HTML, PL/SQL and C# is typically used together with ASP.NET, T-SQL according to our data set. XML and XSD are widely detected in the stacks as well.

Furthermore, in the last part of this chapter, the popularity of the technologies in each functionality type is described by counting the number of systems that are grouped into the *technology stacks* which the technologies are included in. Based on SIG's data set, Java is more than twice as popular as C#. The order of the popularity of the 8 web technologies is: JavaScript, JSP, ASP.NET, FreeMarker, JSF, HTML, Razor and GSP. Moreover, PL/SQL is much more frequently used than T-SQL. And XML is the most widely used technology for the data exchange purpose.

Generally, according to SIG's data set, Java and C# are the most popular general-purpose technologies. And according to the *technology stacks*, Java is always used together with JSP, JSF, FreeMarker, JavaScript, HTML, PL/SQL and C# is typically detected being used together with ASP.NET, T-SQL for the system implementation. Besides, JavaScript, JSP and ASP.NET are the most popular web technologies. While XML and XSD are widely detected from these stacks for adding the data exchange functionality to the systems.

Chapter 4 Relation Between the Technology Selection and the Industry Type

In this Chapter, we are going to visualize the results from Chapter 2 and 3, combining the categories of the system - industry branch with the groups of the technology into one graph to detect the differences of the technology selection among the industry branches.

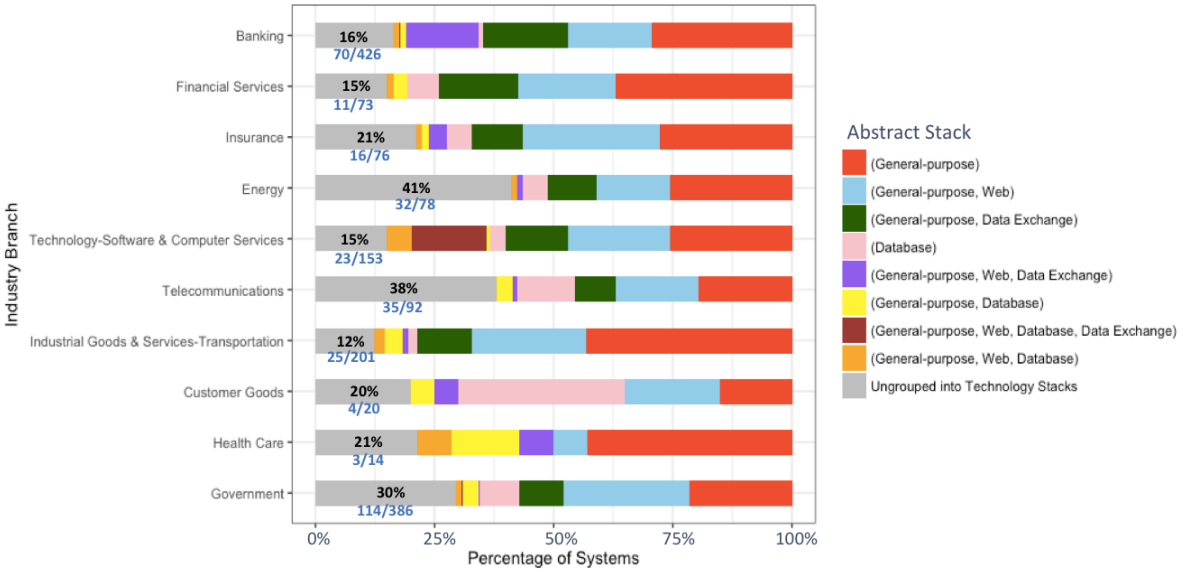


Figure 4.1 System Distribution Among Industry Branches in Abstract Stacks

Figure 4.1 shows the systems distribution among the industry branches in the 8 *abstract stacks* and the percentage of the systems without any *technology stacks*. Energy, Telecommunications and Government industries have the largest percentages of systems that are ungrouped. It means that compared with other industries, these three industries might use more uncommonly used technologies as their dominant technologies while implementing the systems. Besides, according to Figure 4.1, almost every *abstract stack* appears in these 10 industry branches. Nearly all these 8 groups of *Technology Functionality Type Combination* are taken by the systems from all of these industries. It means that the 4 *Technology Functionality Types*: general-purpose technology, web technology, database technology and data exchange technology are widely needed by the systems from all the industries.

4.1 The Use of Technology in Each Industry Branch

For each industry branch, the number of systems that use the technologies from the same functionality type is counted and the proportion of each technology is calculated (the proportion is shown in Appendix D). The proportion of the technologies represents the popularity of these technologies in each industry.

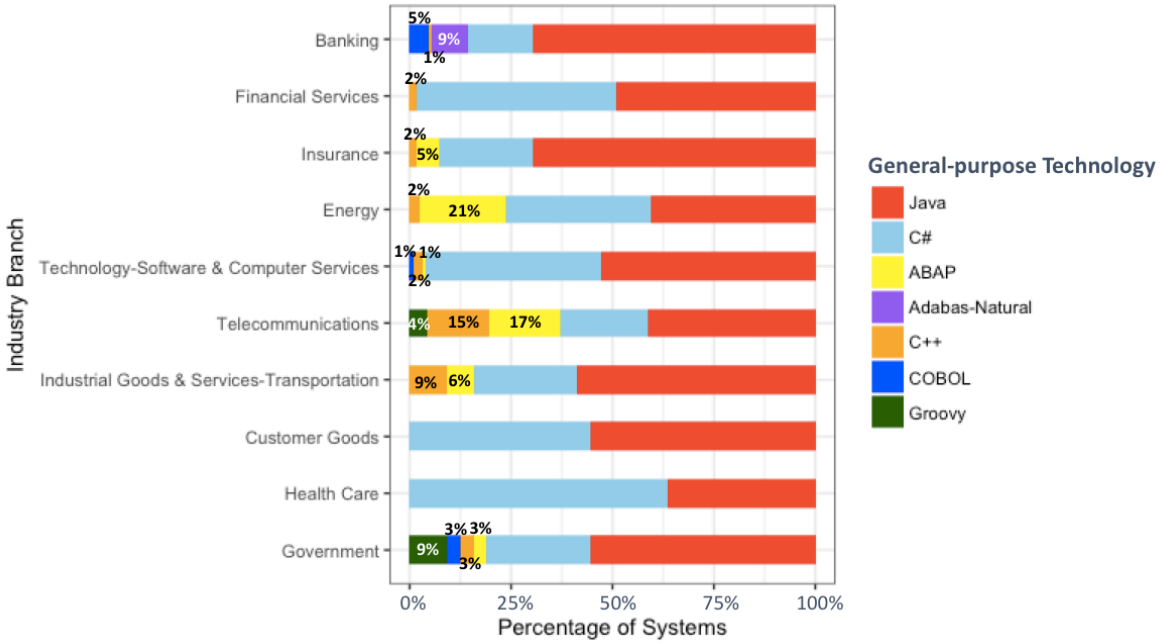


Figure 4.2 The Use of General-purpose Technology in Industry Branches

Table 4.1 The Use of General-purpose Technology (Exclude Java & C#)

Technology	Number of Companies	Number of Systems
C++	17	40
ABAP	12	39
Adabas-Natural	1	31
COBOL	4	25
Groovy	4	24

According to Figure 4.2, all these 10 industries have quite large percentages of Java and C#. Banking is the only industry that use Adabas-Natural, which is known as a minor technology. COBOL, an old technology which was created in the 1950s, is only detected from Banking, Telecommunications and Government industries with very little proportions. While Groovy is only detected in Telecommunications and Government industries. The reason could be that Groovy is a programming language which runs on Java platform, and it is eclipsed by Java to some extent. Besides, according to Table 4.1, among the 1,186 systems from 172 companies, these three technologies are only used by several companies with 20 to 30 systems in total. As for C++ and ABAP, these two technologies are used by 17 companies from 8 different industries and 12 companies from 6 industries respectively, but only a few systems are grouped into the (C++) and (ABAP) *technology stacks*. It means even though these two technologies seem popular among the industries, they are not widely used in each company. The reason could be that in commercial applications, these two technologies require relatively high technical skills for the developers. Therefore, based on our data sets, these five technologies are not as common as Java and C#.

Since Java and C# are the ones with the most frequent uses in every industry, the comparison of the technology among the industries within the General-purpose type will only be conducted between Java and C#.

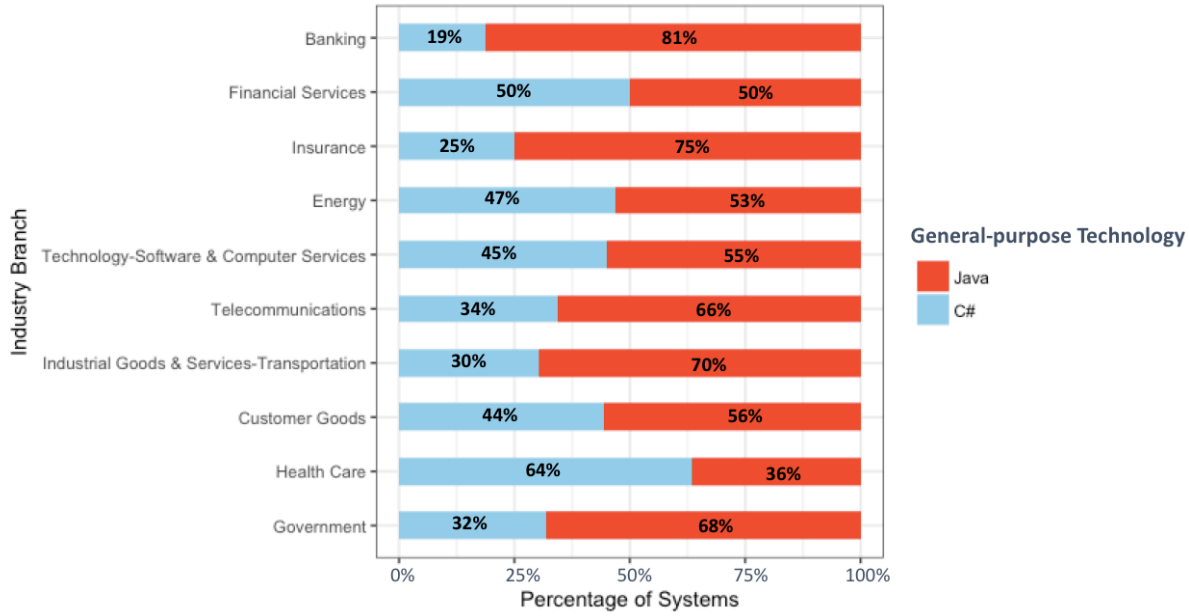


Figure 4.3 The Use of Java & C# in Industry Branches

From Figure 4.3, we can see that in almost all the industries, Java is more popular than C#. While only in the Health Care industry, C# is more commonly used than Java, and in the Financial Services industry, Java and C# share similar popularity.

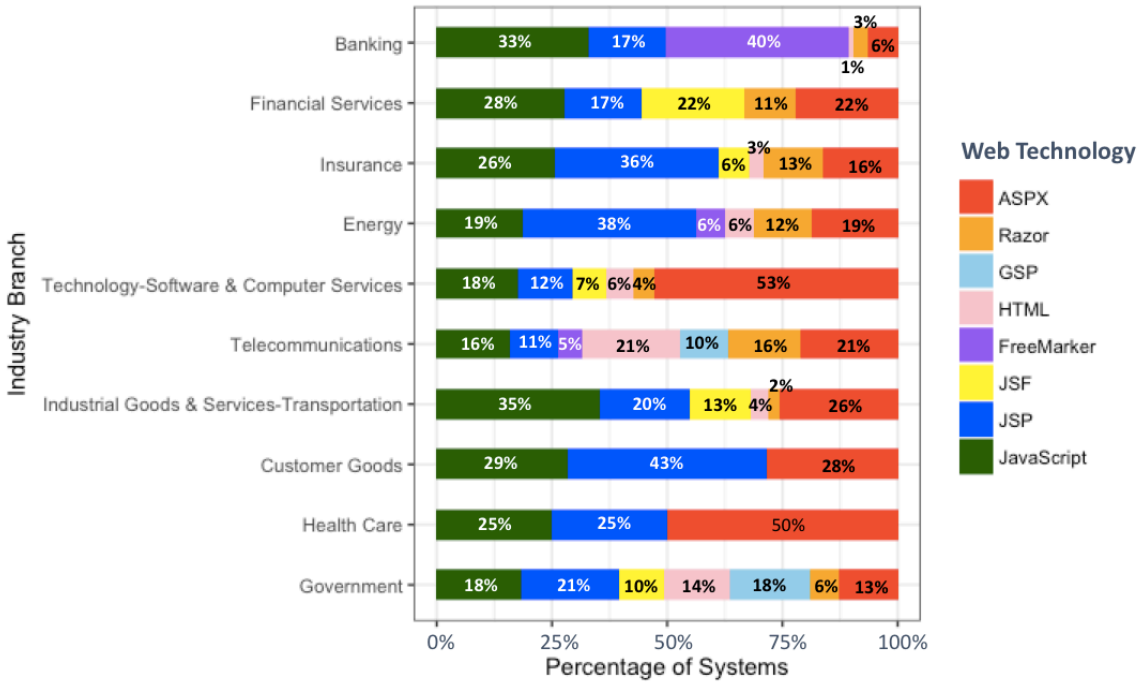


Figure 4.4 The Use of Web Technology in Industry Branches

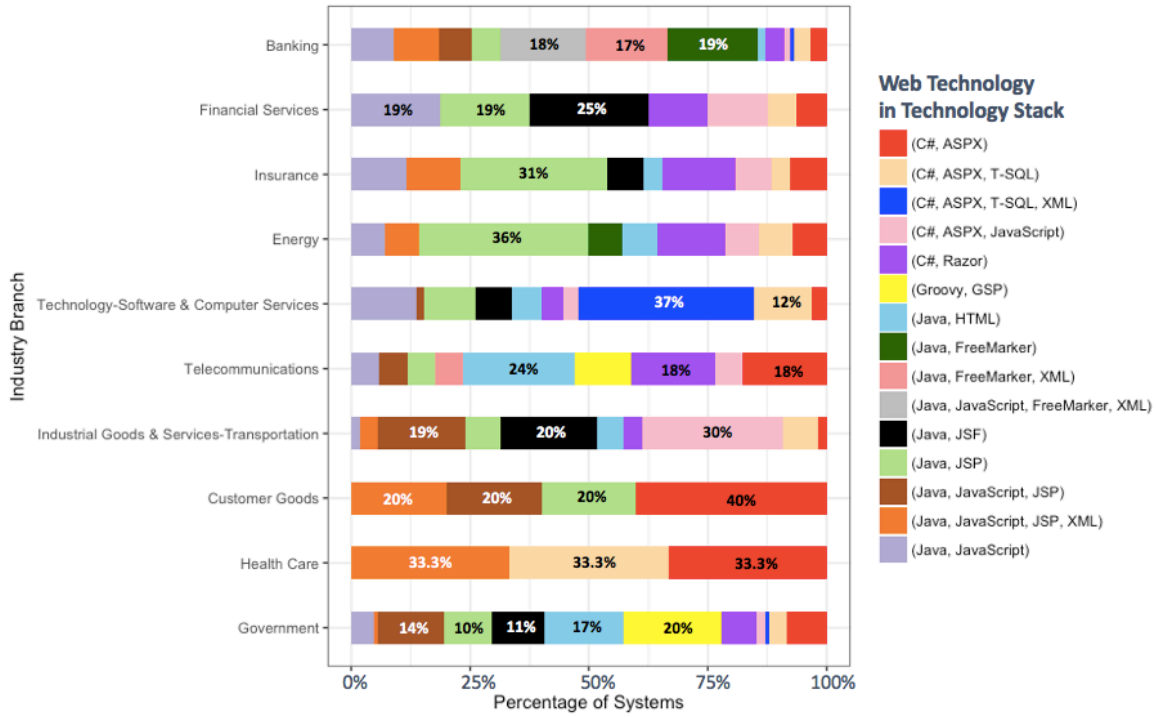


Figure 4.5
The Use of Web Technology Within Technology Stacks in Industry Branches

As it is shown in Figure 4.4 and Figure 4.5, ASP.NET is the most dominant web technologies in Technology-Software & Computer Services and Health Care industries. In these two industries, over 50% systems are grouped into (C#, ASPX), (C#, ASPX, T-SQL) and (C#, ASPX, T-SQL, XML) *technology stacks*. While most industries have large percentages of the systems grouped into the *technology stacks* that contain JSP and JavaScript. These two web technologies frequently exist in the stacks together with Java. FreeMarker seems to be the most commonly used web technology in the Banking industry with many systems grouped into (Java, FreeMarker), (Java, FreeMarker, XML) and (Java, JavaScript, FreeMarker, XML) stacks. However, as it is shown in Table 4.2, FreeMarker is not a popular web technology from the company's perspective, since there are only three companies use it. The systems that select FreeMarker for the system implementation are only from one Banking company. And this technology is not used by the other companies in the Banking industry. Therefore, it is not convincing to get the conclusion that FreeMarker is a popular web technology in Banking industry.

Table 4.2 The Use of FreeMarker

Company	Industry Type of the Company	Number of Systems Use FreeMarker in This Company
Company 12	Banking	79
Company 49	Energy	1
Company 89	Telecommunications	1

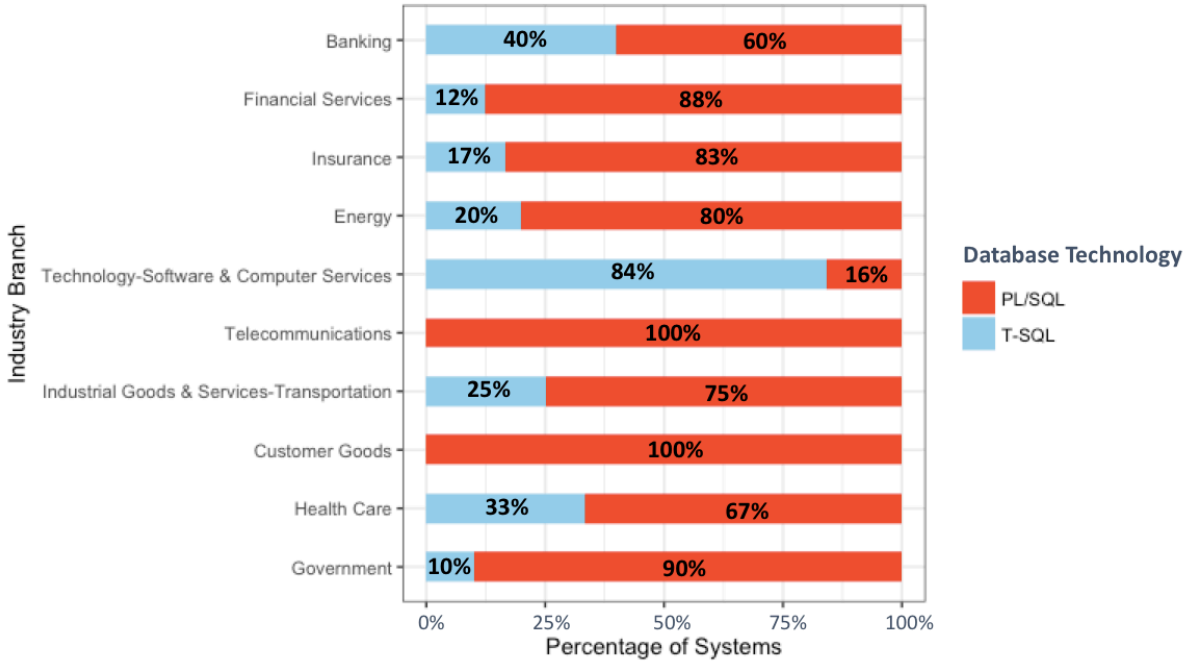


Figure 4.6 The Use of Database Technology in Industry Branches

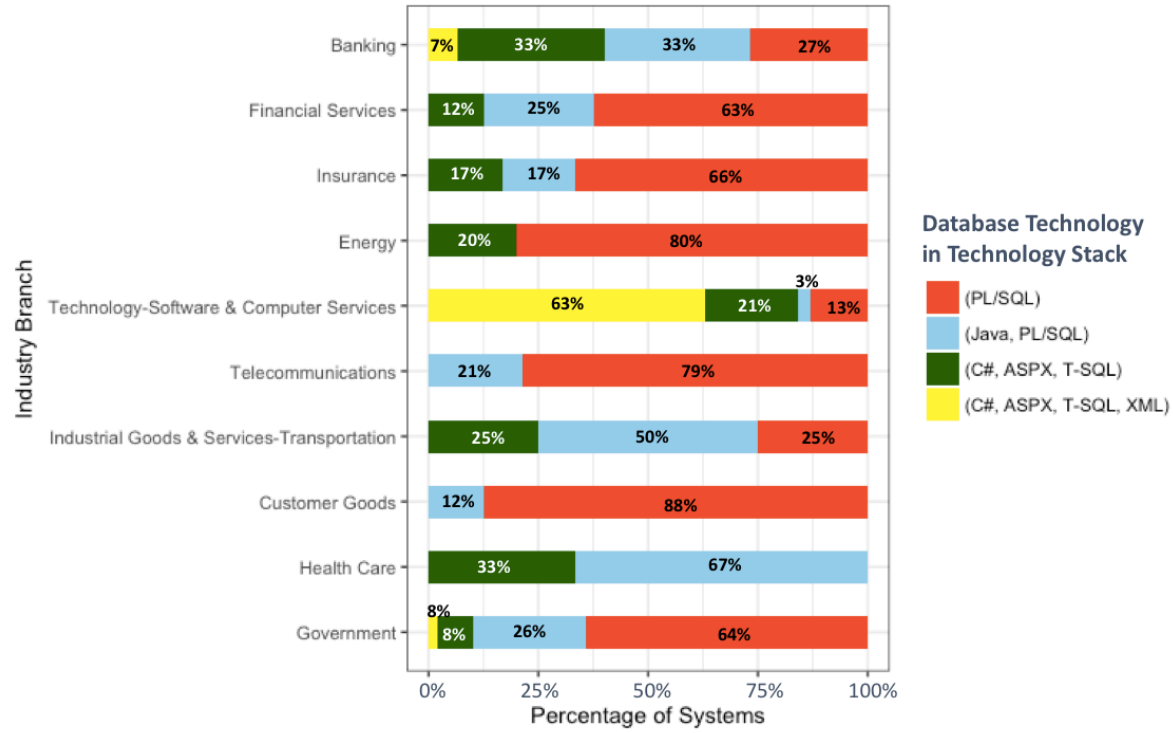


Figure 4.7
The Use of Database Technology Within Technology Stacks in Industry Branches

According to Figure 4.6 and Figure 4.7, except the Technology-Software & Computer Services industry which has more systems using T-SQL than using PL/SQL, other 9 industries use PL/SQL much more frequently. Many systems from the Technology-Software & Computer Services industry are grouped into the *technology stacks* (C#, ASPX, T-SQL) and (C#, ASPX, T-SQL, XML).

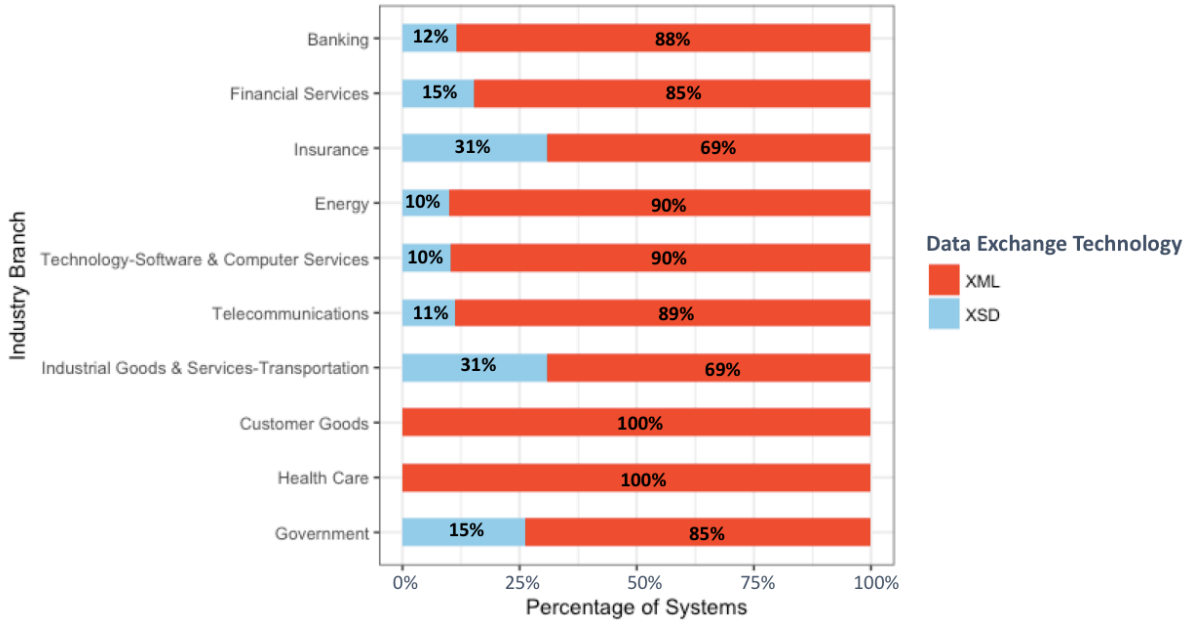


Figure 4.8 The Use of Data Exchange Technology in Industry Branches

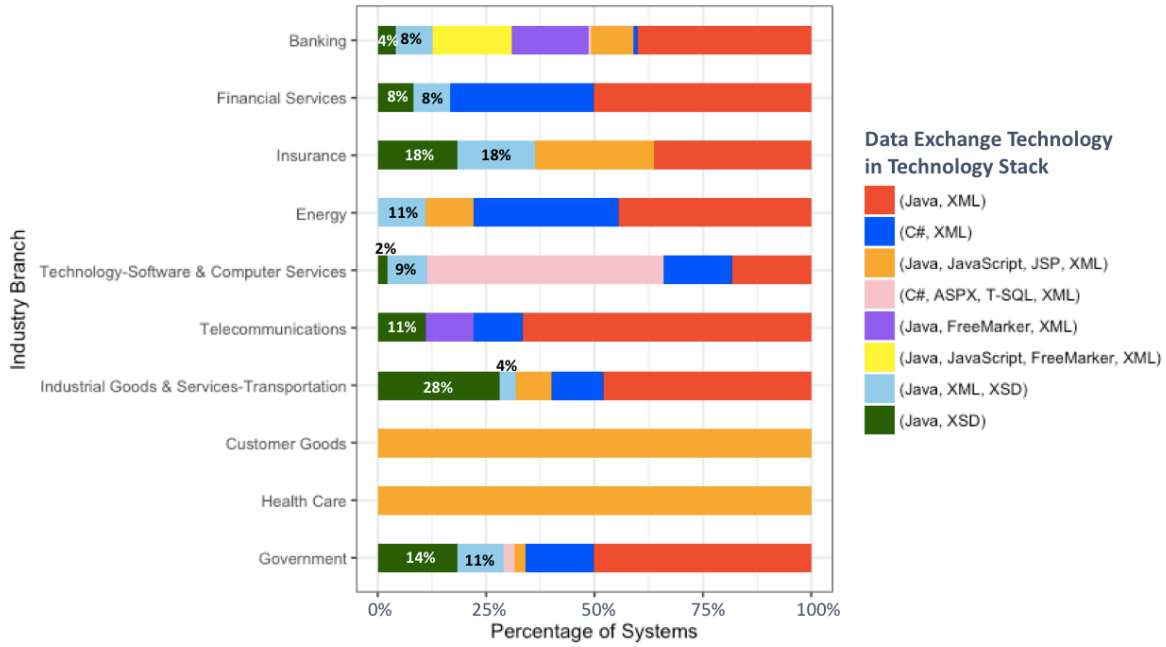


Figure 4.9
The Use of Data Exchange Technology Within Technology Stacks in Industry Branches

Figure 4.8 shows that no matter in which industry, XML is much more frequently used than XSD. There is a large number of systems belonging to (Java, XML) and (C#, XML) *technology stacks*.

4.2 Comparing the Use of Technology Among Industry Branches

The deviations of the technologies' proportion from the average are calculated in order to compare the use of the technologies that are from the same functionality type among all these industry branches. On the basis of Figure 4.3, Figure 4.4, Figure 4.6 and Figure 4.8, the

following four tables, Table 4.3, Table 4.4, Table 4.5 and Table 4.6 are made to compare the technology proportion with the average value. The average proportion of each technology is shown in the last row of the graph and the deviation from the average are calculated for each industry. The colors are used to visualize the differences from the average. The green color on the background of the cells represents the figures that are above the average, and the darker the green color is, the more the figure exceeds the average. Conversely, the red color represents the figures that are below the average. The darker the red color is, the lower the figure is compared with the average. And the white color stands for the figures that are on the average.

Table 4.3 Java & C# Deviation Among Industries

	Java	C#
Banking	20%	-20%
Financial Services	-11%	11%
Insurance	14%	-14%
Energy	-8%	8%
Technology-Software & Computer Services	-6%	6%
Telecommunications	5%	-5%
Industry Goods & Services-Transportation	9%	-9%
Customer Goods	-5%	5%
Health Care	-25%	25%
Government	7%	-7%
Average	61%	39%

It is shown in Table 4.3 that among all these 10 industry branches, the Health Care industry is more in favor of C#, while the Banking industry is more in favor of Java.

Table 4.4 Web Technology Deviation Among Industries

	ASPX	Razor	GSP	HTML	FreeMarker	JSF	JSP	JavaScript
Banking	-19%	-4%	-3%	-4%	35%	-6%	-7%	8%
Financial Services	-3%	4%	-3%	-5%	-5%	16%	-7%	3%
Insurance	-9%	6%	-3%	-2%	-5%	0%	12%	1%
Energy	-6%	5%	-3%	1%	1%	-6%	14%	-6%
Technology-Software & Computer Services	28%	-3%	-3%	1%	-5%	1%	-12%	-7%
Telecommunications	-4%	9%	7%	16%	0%	-6%	-13%	-9%
Industry Goods & Services-Transportation	1%	-5%	-3%	-1%	-5%	7%	-4%	10%
Customer Goods	3%	-7%	-3%	-5%	-5%	-6%	19%	4%
Health Care	25%	-7%	-3%	-5%	-5%	-6%	1%	0%
Government	-12%	-1%	15%	9%	-5%	4%	-3%	-7%
Average	25%	7%	3%	5%	5%	6%	24%	25%

As for the web technologies, Technology-Software & Computer Services and Health Care industries prefer to use ASP.NET much more than the other industries. And the Banking industry uses much more FreeMarker than the other industries. However, according to Table 4.2, the reason is that there is a large number of systems in only one of the Banking company use this technology.

Table 4.5 Database Technology Deviation Among Industries

	PL/SQL	T-SQL
Banking	-16%	16%
Financial Services	12%	-12%
Insurance	7%	-7%
Energy	4%	-4%
Technology-Software & Computer Services	-60%	60%
Telecommunications	24%	-24%
Industry Goods & Services-Transportation	-1%	1%
Customer Goods	24%	-24%
Health Care	-9%	9%
Government	14%	-14%
Average	76%	24%

According to Table 4.5, the Technology-Software & Computer Services industry is more likely to select T-SQL as their dominant database technology compared to other industries.

Table 4.6 Data Exchange Technology Deviation Among Industries

	XML	XSD
Banking	3%	-3%
Financial Services	0%	0%
Insurance	-16%	16%
Energy	5%	-5%
Technology-Software & Computer Services	5%	-5%
Telecommunications	4%	-4%
Industry Goods & Services-Transportation	-16%	16%
Customer Goods	15%	-15%
Health Care	15%	-15%
Government	-11%	11%
Average	85%	15%

Table 4.6 shows that even though XML is the most popular technology for data exchange in every industry, Insurance and Industry Goods & Services-Transportation industries are more in favor of XSD compared with the other industries.

4.3 Results Discussion

Combining the main findings from Section 4.1 and 4.2, the conclusions for this chapter are made from the angle of each industry:

Banking: The Banking industry prefers to use more Java and Java-based web technologies as well as PL/SQL than the C# and C#-based technologies (ASP.NET, T-SQL). Moreover, there is only one company from the Banking industry that has many systems implemented by using FreeMarker. But this technology is not used by the other 14 companies in the Banking industry.
Financial Services: The Financial Services uses more *technology stacks* that contain Java. Besides, JavaScript is the most popular web technology in this industry.
Insurance: The Insurance industry prefers to use more *technology stacks* that contain Java compared to those which contain C#. Besides, JSP is the most popular web technology in this industry. Moreover, it is more likely to use XSD compared with other industries.
Energy: The Energy industry is also in favor of Java-based *technology stacks*. Moreover, compared with other industries, it is more likely to use JSP. However, this industry has the largest percentage of systems that are not grouped into any *technology stacks*, which means

that this industry might use more technologies that are not commonly used compared with other industries.
Technology-Software & Computer Services: Compared with other industries, this industry is much more in favor of ASP.NET and T-SQL. This industry is the most Microsoft-oriented one among all these 10 industries according to our data sets.
Telecommunications: The Telecommunications industry is more prone to using Razor and HTML as the dominant web technologies compared with other industries. Moreover, this industry has the second largest percentage of systems ungrouped into *technology stacks*. It can be inferred that compared with other industries, this industry may have some different preferences on the technology option.
Industry Goods & Services-Transportation: This industry uses more *technology stacks* with Java compared with the stacks with C#. Besides, this industry as well as Insurance industry, are more likely to use XSD compared with other industries.
Customer Goods: This industry has more systems grouped into the *technology stacks* that contain Java, Java-based web technologies and PL/SQL as well. However, it only contains XML as their dominant data exchange technology. The reason could be: There are only 20 data sets categorized into this industry according to SIG's data warehouse.
Health Care: The Health Care industry is the only industry which is detected using more C# than Java. It is also more in favor of ASP.NET than the average of all the industries. However, there are still more systems using PL/SQL instead of T-SQL. The reason could be: There are only 14 systems categorized into this industry, the number of data sets is not large enough to support our conclusion.
Government: This industry is more in preference of Java and Java-based web technologies as well as PL/SQL. It is more likely to use Groovy along with GSP compared with other industries according to the data set from SIG.

Overall, among all these 10 industries, Java and C# are the most popular general-purpose technologies compared to others. Moreover, Java and Java-based web technologies as well as PL/SQL are much more widely used than the *technology stacks* that contain C#, except the Technology-Software & Computer Services industry, which has a larger percentage of systems categorized into the *technology stacks* that contain C#, ASP.NET and T-SQL. Other industries are much more in favor of Java, JavaScript, JSF, JSP and PL/SQL. In General, the most popular web technologies are JavaScript and JSP. Meanwhile, XML is the first option of the technology for data exchange regardless of the industry.

Chapter 5 Conclusions

In this thesis, our research is focused on comparing the use of technology among different industry branches. In order to answer the main research question: To what extent do different industries make different technology decisions for implementing software systems? three sub research questions are set to guide this research in Section 1.2. And in Section 5.1, we are going to summarize the answers to those questions.

5.1 Answers to Research Questions

RQ1. How to classify systems into corresponding industry branches?

By using the Industry Classification Benchmark and conducting the interviews with 14 people working in SIG, 1,519 systems are categorized into the following 10 industries:
- Banking
- Financial Services
- Insurance
- Energy
- Technology – Software & Company Services
- Telecommunications
- Industrial Goods & Services –Transportation
- Customer Goods
- Health Care
- Government

The methodologies used for the system - industry classification are described in Chapter 2, Section 2.1. And the detailed results are shown in Section 2.3. Note that the systems are not equally distributed in these industry branches. There are much more Banking and Government systems compared with the systems in other industries in SIG's data warehouse. 426 systems are grouped into the Banking industry and 386 systems are grouped into the Government industry. However, Health Care and Customer Goods industries only have 14 and 20 systems respectively.

RQ2. Can we find commonly used technologies from these systems?

To find the commonly used technologies, we are trying to group the systems based on their use of technologies. After creating a system grouping model and implementing the algorithm which was written based on the model, 78% systems (1,186 out of 1,519 systems) are grouped into 28 *technology stacks*. The descriptions of the model and the algorithm can be found in Chapter 3, Section 3.3 and 3.4 respectively. The *technology stacks* represent the technology combinations that are commonly used for the system implementation. Each of the 1,186 systems is grouped into only one *technology stack*. Based on these *technology stacks*, 19 technologies in total are extracted from the stacks, which means according to the data set from SIG's data warehouse, these 19 technologies are widely used as the dominant system implementation technologies. Then we categorized these technologies into General-purpose, Web, Database and Data Exchange groups based on the literature review of the functionality type of these technologies. ABAP, Adabas-Natural, C#, C++, COBOL, Groovy and Java are general-purpose technologies which are able to provide multiple functionalities; ASP.NET, FreeMarker, GSP, HTML, JavaScript, JSF, JSP and Razor are web technologies which are

focusing on creating web pages and web applications; PL/SQL and T-SQL are database technologies; XML and XSD are the technologies for data exchange. The definition of each *Technology Functionality Type* as well as the results of the technology categorization are illustrated in Section 3.5.2. Additionally, based on the *Technology Functionality Type*, the 28 *technology stacks* are grouped into 8 *abstract stacks* (the groups of *Technology Functionality Type Combination*). From the *abstract stacks*, we find that almost all the *abstract stacks* contain the general-purpose technologies. And every web technology, database technology or the data exchange technology has to be used together with a general-purpose technology in the *technology stacks*, as it is described in Section 3.5.3. Moreover, according to our data sets, Java and C# are the most frequently used general-purpose technologies. Java always used together with JSP, JSF, FreeMarker, JavaScript, HTML, PL/SQL and C# is typically used together with ASP.NET, T-SQL for the system implementation. While XML and XSD are frequently detected in these stacks for adding the data exchange functionality to the systems.

RQ3. What is the relation between the results from sub research questions 1 and 2?

Generally, among all these 10 industries, Java and C# are the most popular general-purpose technologies compared to others. Moreover, Java and Java-based web technologies as well as PL/SQL are much more widely used than the *technology stacks* that contain C#, except the Technology-Software & Computer Services industry, which has a larger percentage of systems categorized into the *technology stacks* that contain C#, ASP.NET and T-SQL. Other industries are much more in favor of Java, JavaScript, JSF, JSP, FreeMarker and PL/SQL. In General, the most popular web technologies are JavaScript and JSP. Meanwhile, XML is the first option of the technology for data exchange regardless of the industry. These results are generated and summarized from the graphs and tables in Chapter 4.

5.2 Threats to Validity

The threats to validity can be divided into three categories: Construct Validity, Internal Validity and External Validity (Perry et al., 2000).

5.2.1 Construct Validity

Do the variables and hypotheses of our study accurately model the research questions?

Methods of collecting the technologies for the industry comparison. As it is mentioned in Section 1.2, to prevent the distinct technology selections that might be caused by the developer's or the project's preferences, the technologies that are only detected in a few systems are excluded in this research. We make an assumption at the beginning of the research that there are some technologies that are only widely used by some industries, but not frequently used by the others. Therefore, in Chapter 3, only the technology combinations that are relatively commonly used are collected for the further industry comparison. The threshold for the "*Lower Limit Number of Systems Per Stack*" which is set during the system grouping process makes sure that only the commonly used technology combinations are selected. Based on the threshold, around 80% systems are grouped into 28 *technology stacks* (as it is described in Section 3.5). The commonly used technologies are collected from these *technology stacks* and there are 19 technologies altogether. The industry comparison is conducted within these 28 *technology stacks* and 19 technologies in our research.

However, if there is no assumption made at the beginning of the research, it means, if the technologies or the technology combinations that are only detected in a few systems are included before the industry comparison, all the technologies as well as the technology combinations will be collected for the comparison without creating the *technology stacks* in advance. However, this method will increase the complexity of the industry comparison work. Because during the comparison process, each technology as well as the technology comparison should be marked with the "*System*", the "*Company*" and the "*Industry*" labels, which are used for counting the frequency of the technology combinations among the systems, the companies as well as the industries. After counting the frequency of each label and setting the threshold, the systems that are not frequently used by any industries are filtered out. If the same method is used to set the threshold, the final results will probably be similar to what we get through creating the *technology stacks* for the technology collection before the industry comparison.

5.2.2 Internal Validity

Are the changes in the dependent variables safely attributed to the changes in the independent variables?

Assumption of the threshold for the lower limit number of systems per *technology stack*. In our research, we use the formula: $K \approx \sqrt{(\frac{n}{2})}$, where n is the number of data points, in order to find the most suitable number of *technology stacks* for our data sets. And based on it, the threshold for the "*Lower Limit Number of Systems Per Stack*" is set to 23. However, if we set different thresholds for the "*Lower Limit Number of Systems Per Stack*", we will get different number of *technology stacks* and the total number of technologies that are included in the stacks will be different as well. If the threshold is lower than 23, we will get more than 28 *technology stacks*, and there will probably be more technologies altogether from the *technology stacks*. While if the threshold is higher than 23, less *technology stacks* as well as the total number of technologies will be extracted.

Unequal number of systems in the industry branches. Because our data sets are collected from SIG's data warehouse and according to these data sets, there are 426 and 386 systems in Banking and Government industries, but only 14 and 20 systems in Health Care and Customer Goods industries. The systems from SIG's data warehouse are unequally distributed among the industry branches. The industries with larger number of systems have more influence on the popularity order of the technologies in Section 3.5.4. Additionally, since there are a large number of Banking and Government systems, some distinct technologies are only detected in these industry, like COBOL and Groovy, as it is described in Section 4.1. If the system numbers of other industries increase, these technologies might be detected in other industries as well.

Methods of defining the dominant technologies for each system. As it is illustrated in Section 3.1, the importance of the technologies for each system are measured based on the volume proportion. The technology with the largest volume proportion is regarded as the most dominant one while the technology with the smallest volume proportion is the least dominant one. For each technology in a certain system, the volume proportion is calculated by using the volume of that technology divided by the aggregate technologies' volume. And SIG expresses volume as rebuild value in man years. If different methods are used to measure the importance of the technologies, the dominant technologies that are selected for each system could be different. For instance, if we make interviews with the system analysts to collect the dominant technologies based on their perspectives, the dominant technologies collected from the systems will be different and then different *technology stacks* will be created.

5.2.3 External Validity

Can the study results be generalized to settings outside the study?

Generalization to another data set. Since the data sets used by this research are only collected from SIG's data warehouse, the results are more SIG-oriented and cannot fully represent the worldwide technology usage. In SIG's data warehouse, there are much more Banking and Government systems compared to other industries. If we enlarge the data sets, there will probably be more systems categorized into the other industries. The total number of technologies used by these industries will increase as well. For instance, there are only 14 systems categorized into the Health Care industry, and the only general-purpose technologies that are extracted from the *technology stacks* in this industry are Java and C# (as it described in Figure 4.2). If the data sets are extent, there probably will be more systems in the Health Care industry, and there might be some new *technology stacks* that contain Python or Ruby as the general-purpose technologies created through the same algorithm for instance. The results for analyzing the use of technologies in each industry might be different from the results that are obtained from this research.

5.3 Future Work

There are several directions to which our research can be extended. Adding the system functionality type into the analysis to detect the functional requirements' influence on the technology selection and extending the data set to find more technology combinations that are used for the system implementation are the most valuable two directions.

Detect the functional requirements' influence on the technology selection

Our research is focusing on the technology usage comparison among different industry branches to explore the relation between the industry's preference and the technology selection. However, since the system has a lot of attributes, the industry type is just one attribute that can be easily get with knowing the name of the company that the system belongs to, the functionality type of the system is also an interesting attribute. Since the systems are implemented for a certain or multiple functional purposes. For instance, ERP systems are focusing on automating and integrating companies' business processes, some systems are implemented for providing the interface between a human and several systems or services, while some systems with predetermined algorithms stored in them are able to analyze the data to provide decision support. The results from comparing the technology usage among different functionality types are also valuable to the scientific world. And many methodologies described in this research can be reused then, like the "*Data Modelling*", "*Interview*", "*Double Checks*", "*Hierarchical Classification*" methodologies described in Chapter 2 and the approach for grouping the systems based on the use of technology in Chapter 3.

It is worth noting that the system - functionality type classification work is much more tough than the system - industry branch classification work. The first reason is that the system - industry branch classification work can be transformed to the company - industry branch classification work. After knowing the industry branches of the companies, a lot of systems that belong to these companies are classified into the industry branches. However, since the systems are probably working on different functional purposes, the interviews for the system -

functionality type classification should be conducted with the interviewees who are familiar with the systems and the systems should be categorized individually. Apart from the interviews for the system - industry branch classification, we made interviews for the system - functionality type classification with the same 14 interviewees as well. As is it shown in Appendix E, the Business Application Classification Benchmark (Hoekstra, 2015) is used for the interviews. There are 102 data sets collected from the interviews, which means the system - functionality type classification is made for 102 systems. However, as it is depicted in Figure 5.1, through the "*Double Checks*" and "*Hierarchical Classification*" methodologies, only 57% of the data sets can be used for further analysis. It means that according to two interviewees' opinions, a large percentage of the systems are categorized into totally different groups. The second reason that makes the system - functionality type classification work tough could be: Many systems provide multiple functionalities and they are not limited to only one category. Therefore, to match the benchmark with the suitable data set should be the first step of the system - functionality type classification work. With an overview of the system functionality at first, to match the benchmark with the data sets, some modifications on the benchmark within a reasonable scale are allowed.

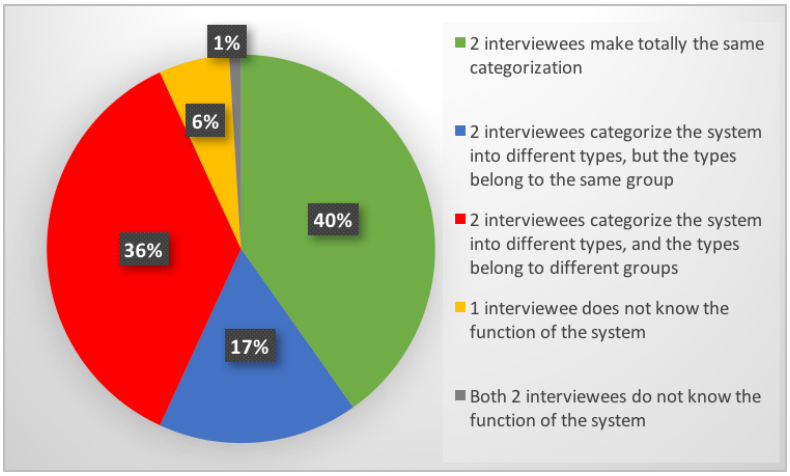


Figure 5.1 Results from System Functionality Classification Interview

Extend the Technology Data Set

In this research, 28 *technology stacks* are created and 19 technologies are collected from these stacks. Based on these *technology stacks*, some frequently used technology combinations are detected, as it is described in Section 3.5.3. For example, the web technologies, JSP and JSF are always used together with Java, while ASP.NET is a C#-based web technology. However, there are more technologies that are being used for the software implementation. The information exists in many online resources, like GitHub, Stack Overflow and so on. Figure 5.2 shows the technology landscape from Stack Overflow. On the basis of the functionality type, most of these technologies can be categorized into the four categories, General-purpose, Web, Database and Data Exchange. Moreover, according to the findings from the *abstract stacks* in this research, web technologies, database technologies and the technologies for data exchange are always used together with a general-purpose technology. The technology relations, like which web technology is always used together with which general-purpose technology can be collected from some literature reviews then. If the *technology stacks* are created in this way (only through the literature review), more technology combinations which can be theoretically used for implementing the systems will be collected. These *technology*

stacks can be verified in any data sets in order to discover the technology combinations that are commonly used for the system implantation in the real world.

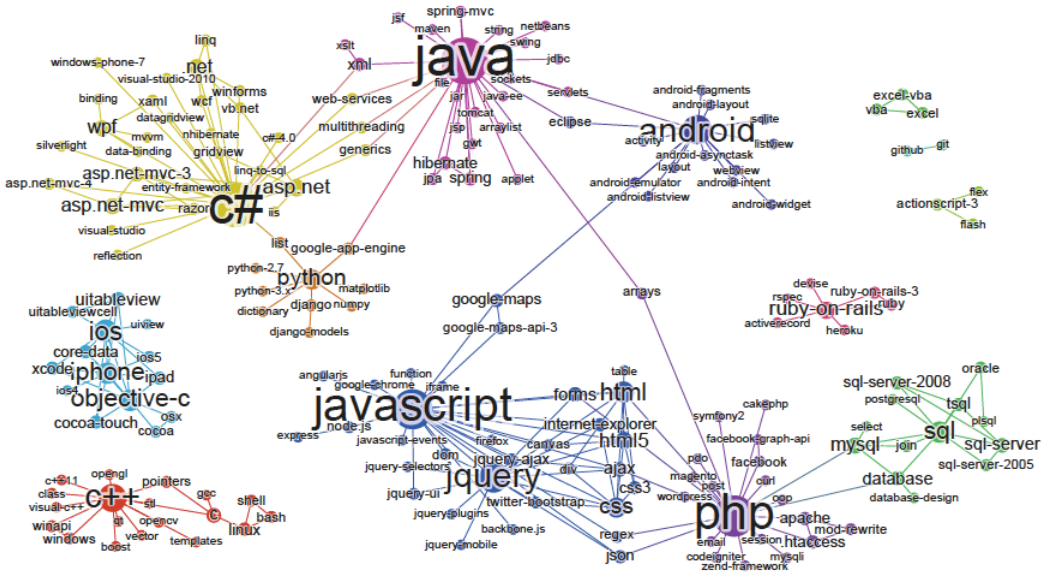


Figure 5.2 Technology Landscape from Stack Overflow
(Chen et al., 2016)

Bibliography

ABAP. n.d. *In Wikipedia*. Retrieved June 26, 2017, from: https://en.wikipedia.org/wiki/ABAP .

ADABAS. n.d. *In Wikipedia*. Retrieved June 26, 2017, from: https://en.wikipedia.org/wiki/ADABAS .

Agrawal, R., Gehrke, J., Gunopulos, D., & Raghavan, P. (2005). Automatic subspace clustering of high dimensional data. *Data Mining and Knowledge Discovery*, *11*(1), 5-33.

Arranga, E. C., & Coyle, F. P. (1996). *Object-oriented COBOL*. Cambridge University Press.

ASP.NET. n.d. *In Wikipedia*. Retrieved June 26, 2017, from: https://en.wikipedia.org/wiki/ASP.NET .

ASP.NET Razor. n.d. *In Wikipedia*. Retrieved June 26, 2017, from: https://en.wikipedia.org/wiki/ASP.NET_Razor .

Bouwers, E. M. (2013). Metric-based Evaluation of Implemented Software Architectures.

Chen, C., & Xing, Z. (2016). Mining technology landscape from stack overflow. In Proceedings of the 10th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (p. 14). ACM.

Chen, P. P. S. (1976). The entity-relationship model—toward a unified view of data. *ACM Transactions on Database Systems (TODS)*, *1*(1), 9-36.

ComputerWeekly. (2017). Adabas continues to play a vital role for installed base [Web log post]. Retrieved June 24, 2017, from: http://www.computerweekly.com/news/2240065879/Adabas-continues-to-play-a-vital-role-for-installed-base .

C Sharp (programming language). n.d. *In Wikipedia*. Retrieved June 26, 2017, from: https://en.wikipedia.org/wiki/C_Sharp_(programming_language) .

C# Corner. (2017). What Can C# Do for You [Web log post]. Retrieved June 26, 2017, from: http://www.c-sharpcorner.com/UploadFile/mahesh/what-can-C-Sharp-do-for-you/ .

Datta, A., & Thomas, H. (1999). The cube data model: a conceptual model and algebra for on-line analytical processing in data warehouses. *Decision Support Systems*, *27*(3), 289-301.

Dorsey, P. (2000). Top 10 reasons why systems projects fail. *Retrieved February*, *10*, 2005.

Dummies. (2017). What Is PL/SQL Good For? [Web log post]. Retrieved June 26, 2017, from: http://www.dummies.com/programming/sql/what-is-plsql-good-for/ .

Fred, A. L., & Jain, A. K. (2005). Combining multiple clusterings using evidence accumulation. *IEEE transactions on pattern analysis and machine intelligence*, *27*(6), 835-850.

FreeMarker. n.d. *In Wikipedia*. Retrieved June 26, 2017, from: https://en.wikipedia.org/wiki/FreeMarker .

FTSE Russell. (2012). Industry Classification Benchmark [Web log post]. Retrieved March 14, 2017, from: http://www.icbenchmark.com/ .

FTSE Russell. (2016). Industry Classification Benchmark (Equity) [Web log post]. Retrieved March 14, 2017, from: http://www.ftse.com/products/downloads/icb_rules.pdf .

Gauch Jr, H. G., & Whittaker, R. H. (1981). Hierarchical classification of community data. *The Journal of Ecology*, 537-557.

Glass, R. L. (1997). Cobol-A contradiction and an enigma. *Communications of the ACM*, *40*(9), 11-14.

Gordon, A. D. (1987). A review of hierarchical classification. *Journal of the Royal Statistical Society. Series A (General)*, 119-137.

Groovy (programming language). n.d. *In Wikipedia*. Retrieved June 26, 2017, from: https://en.wikipedia.org/wiki/Groovy_(programming_language) .

Hall, A. (2017). How to Pick the Right Programming Language [Web log post]. Retrieved March 14, 2017, from: http://mashable.com/2012/07/11/developer-programming-languages/#AfmBrELDKsqf .

Hejlsberg, A., Wiltamuth, S., & Golde, P. (2003). *C# language specification*. Addison-Wesley Longman Publishing Co., Inc..

Hoekstra, F. H. (2015). Functionality based business application classification (Master thesis). Retrieved March 16, 2017, from: http://liacs.leidenuniv.nl/assets/Masterscripties/ICTiB/2015-2016/Hoekstra-Freek.pdf .

HTML. n.d. *In Wikipedia*. Retrieved June 26, 2017, from: https://en.wikipedia.org/wiki/HTML .

Hunt, G. C., & Larus, J. R. (2007). Singularity: rethinking the software stack. *ACM SIGOPS Operating Systems Review*, *41*(2), 37-49.

Jain, A. K., Murty, M. N., & Flynn, P. J. (1999). Data clustering: a review. *ACM computing surveys (CSUR)*, *31*(3), 264-323.

Java (programming language). n.d. *In Wikipedia*. Retrieved June 26, 2017, from: https://en.wikipedia.org/wiki/Java_(programming_language) .

JavaScript. n.d. *In Wikipedia*. Retrieved June 26, 2017, from: https://en.wikipedia.org/wiki/JavaScript .

JavaServer Faces. n.d. *In Wikipedia*. Retrieved June 26, 2017, from: https://en.wikipedia.org/wiki/JavaServer_Faces .

JavaServer Pages. n.d. *In Wikipedia*. Retrieved June 26, 2017, from: https://en.wikipedia.org/wiki/JavaServer_Pages .

Kaur, R., & Sengupta, J. (2013). Software process models and analysis on failure of software development projects. *arXiv preprint arXiv:1306.1068*.

Kishor, N. R. (2014). International Journal of Advance Research in Computer Science and Management Studies. *International Journal*, *2*(3).

Knuth, D. E. (1997). The Art of Computer Programming. Volume 1: Fundamental Algorithms. Volume 2: Seminumerical Algorithms. *Bull. Amer. Math. Soc*.

Kodinariya, T. M., & Makwana, P. R. (2013). Review on determining number of Cluster in K-Means Clustering. *International Journal*, *1*(6), 90-95.

Lawlis, P. K. (1997). Guidelines for choosing a computer language: Support for the visionary organization. *Defence Information Systems Agency*, 1-14.

Levene, M., & Loizou, G. (2003). Why is the snowflake schema a good data warehouse design? *Information Systems*, *28*(3), 225-240.

Mandal, A., & Pal, S. C. Identifying the Reasons for Software Project Failure and Some of their Proposed Remedial through BRIDGE Process Models.

Merriam-Webster. (2017). Definition of SYSTEM [Web log post]. Retrieved March15, 2017, from: https://www.merriam-webster.com/dictionary/system .

Parsons, L., Haque, E., & Liu, H. (2004). Subspace clustering for high dimensional data: a review. *Acm Sigkdd Explorations Newsletter*, *6*(1), 90-105.

Perry, D. E., Porter, A. A., & Votta, L. G. (2000, May). Empirical studies of software engineering: a roadmap. In *Proceedings of the conference on the future of Software engineering* (pp. 345-355). ACM.

PL/SQL. n.d. *In Wikipedia*. Retrieved June 26, 2017, from: https://en.wikipedia.org/wiki/PL/SQL .

Quora. (2017). What Is the Purpose of Groovy and Grails [Web log post]. Retrieved June 26, 2017, from: https://www.quora.com/What-is-the-purpose-of-Groovy-and-Grails .

Quora. (2017). What Can Java Programming Do [Web log post]. Retrieved June 26, 2017, from: https://www.quora.com/What-can-Java-programming-do .

Quora. (2017). What Is C++ Used For [Web log post]. Retrieved June 26, 2017, from: https://www.quora.com/What-is-C++-used-for .

Sarstedt, M., & Mooi, E. (2014). A concise guide to market research. *The Process, Data, and*.

Saunders, M. N. (2011). *Research methods for business students, 5/e*. Pearson Education India.

Shojaee, H. (2007). Why Choosing the Right Technology Matters? [Web log post]. Retrieved March15, 2017, from: https://blog.axosoft.com/2007/12/31/why-choosing-the-right-technology-matters/ .

Srinivasan, K. (2017). Introduction to Groovy Server Pages (GSP) [Web log post]. Retrieved June 26, 2017, from: http://javabeat.net/introduction-to-groovy-server-pages-gsp/ .

Stack Overflow. (2017). What Can JavaScript Do? [Web log post]. Retrieved June 26, 2017, from: https://stackoverflow.com/questions/3387603/what-can-javascript-do .

Stack Overflow. (2017). When Should XSD Files Be Used? [Web log post]. Retrieved June 26, 2017, from: https://stackoverflow.com/questions/6487171/when-should-xsd-files-be-used .

Stroustrup, B. (2013). *The C++ programming language*. Pearson Education.

Transact-SQL. n.d. *In Wikipedia*. Retrieved June 26, 2017, from: https://en.wikipedia.org/wiki/Transact-SQL .

Tungare, M. (2000). Active Server Pages.

Tutorialspoint. (2017). SAP Programming Language (ABAP) [Web log post]. Retrieved June 23, 2017, from: https://www.tutorialspoint.com/sap/sap_programming_language.htm .

Wikibooks. (2017). XML – Managing Data Exchange/ Introduction to XML [Web log post]. Retrieved June 26, 2017, from: https://en.wikibooks.org/wiki/XML_-_Managing_Data_Exchange/Introduction_to_XML .

XML. n.d. *In Wikipedia*. Retrieved June 26, 2017, from: https://en.wikipedia.org/wiki/XML .

XML Schema (W3C). n.d. *In Wikipedia*. Retrieved June 26, 2017, from: https://en.wikipedia.org/wiki/XML_Schema_(W3C) .

Appendix A. Industry Classification Benchmark

Industry Classification Benchmark

Industry(Supersector)		Sector		Subsector	
0500	Oil & Gas	0530	Oil & Gas Producers	0533	Exploration & Production
				0537	Integrated Oil & Gas
		0570	Oil Equipment, Services & Distribution	0573	Oil Equipment, Services & Distribution
				0577	Pipelines
		0580	Alternative Energy	0583	Renewable Energy Equipment
				0587	Alternative Fuels
1300	Chemicals	1350	Chemicals	1353	Commidity Chemicals
				1357	Specialty Chemicals
1700	Basic resources	1730	Forestry & Paper	1733	Forestry
				1737	Paper
		1750	Industrial Metals & Mining	1753	Aluminum
				1755	Nonferrous Metals
				1757	Iron & Steel
		1770	Mining	1771	Coal
				1773	Diamonds & Gemstones
				1775	General Mining
				1777	Gold Mining
				1779	Platinum & Precious Metals
2300	Construction & Materials	2350	Construction & Materials	2353	Building Materials & Fixtures
				2357	Heavy Construction
2700	Industrial Goods & Services	2710	Aerospace & Defense	2713	Aerospace
				2717	Defense
		2720	General Industrials	2723	Containers & Packaging
				2727	Diversified Industrials
		2730	Electronic & Eletrical Equipment	2733	Electrical Components & Equipment
				2737	Electronic Equipment
		2750	Industrial Engineering	2753	Commercial Vehicles & Trucks
				2757	Industrial Machinery
		2770	Industrial Transportation	2771	Delivery Services
				2773	Marine Transportation
				2775	Railroads
				2777	Transportation Services
				2779	Trucking
		2790	Support Services	2791	Business Support Services
				2793	Business Training & Employment Agencies
				2795	Financial Administration
				2797	Industrial Suppliers
				2799	Waste &Diposal Services
3300	Automobiles & Parts	3350	Automobiles & Parts	3353	Automobiles
				3355	Auto Parts
				3357	Tires
3500	Food & Berverage	3530	Beverages	3533	Brewers
				3535	Distillers & Vintners
				3537	Soft Drinks
		3570	Food Producers	3573	Farming & Fishing
				3577	Food Products
3700	Personal & Household Goods	3720	HouseholdGoods & Home Construction	3722	Durable Household Products
				3724	Nondurable Household Products
				3726	Furnishings
				3728	Home Construction
		3740	Leisure Goods	3743	Consumer Electronics
				3745	Recreational Products
				3747	Toys

3700	Personal & Household Goods	3760	Personal Goods	3763	Clothing & Accessories
				3765	Footwear
				3767	Personal Products
		3780	Tobacco	3785	Tobacco
4500	Health Care	4530	Health Care Equipment & Services	4533	Health Care Providers
				4535	Medical Equipment
				4537	Medical Supplies
		4570	Pharmaceuticals & Biotechnology	4573	Biotechnology
				4577	Pharmaceuticals
5300	Retail	5330	Food & Drug Retailers	5333	Drug Retailers
				5337	Food Retailers & Wholesalers
		5370	General Retailers	5371	Apparel Retailers
				5373	Broadline Retailers
				5375	Home Improvement Retailers
				5377	Specialized Consumer Services
				5379	Specialty Retailers
5500	Media	5550	Media	5553	Broadcasting & Entertainment
				5555	Media Agencies
				5557	Publishing
5700	Travel & Leisure	5750	Travel & Leisure	5751	Airelines
				5752	Gambling
				5753	Hotels
				5755	Recreational Sevices
				5757	Restaurants & Bars
				5759	Travel & Tourism
6500	Telecommunications	6530	Fixed Line Telecommunications	6535	Fixed Line Telecommunications
		6570	Mobile Telecommunications	6575	Mobile Telecommunications
7500	Utilities	7530	Electricity	7535	Conventional Electricity
				7537	Alternative Electricity
		7570	Gas, Water & Multiutilities	7573	Gas Distribution
				7575	Multiutilities
				7577	Water
8300	Banks	8350	Banks	8355	Banks
8500	Insurance	8530	Nonlife Insurance	8532	Full Line Insurance
				8534	Insurance Brokers
				8536	Property & Casualty Insurance
				8538	Reinsurance
		8570	Life Insurance	8575	Life Insurance
8600	Real Estate	8630	Real Estate Investment & Services	8633	Real Estate Holding & Development
				8637	Real Estate Services
		8670	Real Estate Investment Trusts	8671	Industrial & Office REITs
				8672	Retail REITs
				8673	Residential REITs
				8674	Diversified REITs
				8675	Specialty REITs
				8676	Mortgage REITs
				8677	Hotel & Lodging REITs
8700	Financial Services	8770	Financial Services	8771	Asset Managers
				8773	Consumer Finance
				8775	Specialty Finance
				8777	Investment Services
				8779	Mortgage Finance
		8980	Equity Investment Instruments	8985	Equity Investment Instruments
		8990	Nonequity Investment Instruments	8995	Nonequity Investment Instruments
9500	Technology	9530	Software & Computer Services	9533	Computer Services
				9535	Internet
				9537	Software
		9570	Technology hardware & Equiment	9572	Computer Hardware
				9574	Electronic Office Equiment
				9576	Semiconductors
				9578	Telecommication Equipment
11100	Government	11110	Government	11111	Government

(FTSE Russell, 2012)

Note: The "Supersector" layer is set as the "Industry" layer in our research compared with the original benchmark. Thus, the four-layer benchmark is transformed to the three-layer benchmark.

Appendix B. List of the Technology from Technology Stacks

Technology	Definition	Functionality Description	Related Technology Stack
ABAP	Advanced Business Application Programming. It is a high-level programming language created by SAP SE. (Wikipedia, 2016)	It can be used for the development of application programs with multiple specific functions including: - Reports - Module Pool Programming - Interfaces - Forms - Data conversions - User Exists & BADI (Business Add-In) (Tutorialspoint, 2017)	(ABAP)
Adabas-Natural	It is an acronym for Adaptable Data Base System. (Wikipedia, 2012)	Adabas is a database management system for IBM mainframes, Vax hardware, Unix and Windows. (ComputerWeekly, 2017)	(Adabas-Natural)
ASPX	ASP is an acronym of Active Server Pages. It is an open, compile-free application environment in which you can combine HTML, scripts, and reusable ActiveX server components to create dynamic and powerful Web-based business solutions (Tungare, 2000).	The type of ASP is the "Web Application Framework". (Wikipedia, 2015)	(C#, ASPX) (C#, ASPX, JavaScript) (C#, ASPX, T-SQL) (C#, ASPX, T-SQL, XML)
C#	It is a programming language that is designed for building a variety of applications that run on the .NET Framework. C# is simple, powerful, type-safe, and object-oriented. (Hejlsberg et al., 2003)	C# can be used to write Windows clients applications, Web applications, Mobile apps, Enterprise software, backend and service-oriented applications. (C# Corner, 2017)	(C#) (C#, Razor) (C#, XML) (C#, ASPX) (C#, ASPX, T-SQL) (C#, ASPX, T-SQL, XML)
C++	C++ is a general-purpose programming language. It was designed with a bias toward system programming and embedded, resource-constrained and large systems, with performance, efficiency and flexibility of use as its design highlights. (Stroustrup, 2013)	C++ is used nearly everywhere for everything, including: - System programming (operating systems, device drivers, database engines, embedded, Internet of Things, etc.) - Numerical and scientific computing - Web development - Desktop applications - … (Quora, 2017)	(C++)
COBOL	It is an acronym for Common Business-Oriented Language. It is a compiled English-like computer programming language designed for business use. (Arranga et al., 1996)	Its roots lie in: - Accessing data - Business computing - File handling - Batch transaction processing - Reports generating (Glass, 1997)	(COBOL)
FreeMarker	FreeMarker is a Java-based Template Engine. It is often used for generating HTML web pages, source code, configuration files or E-mails. (Wikipedia, 2014)	It is a "Template Engine" for generating web pages.	(Java, FreeMarker) (Java, FreeMarker, XML Framework) (Java, JavaScript, FreeMarker, XML)

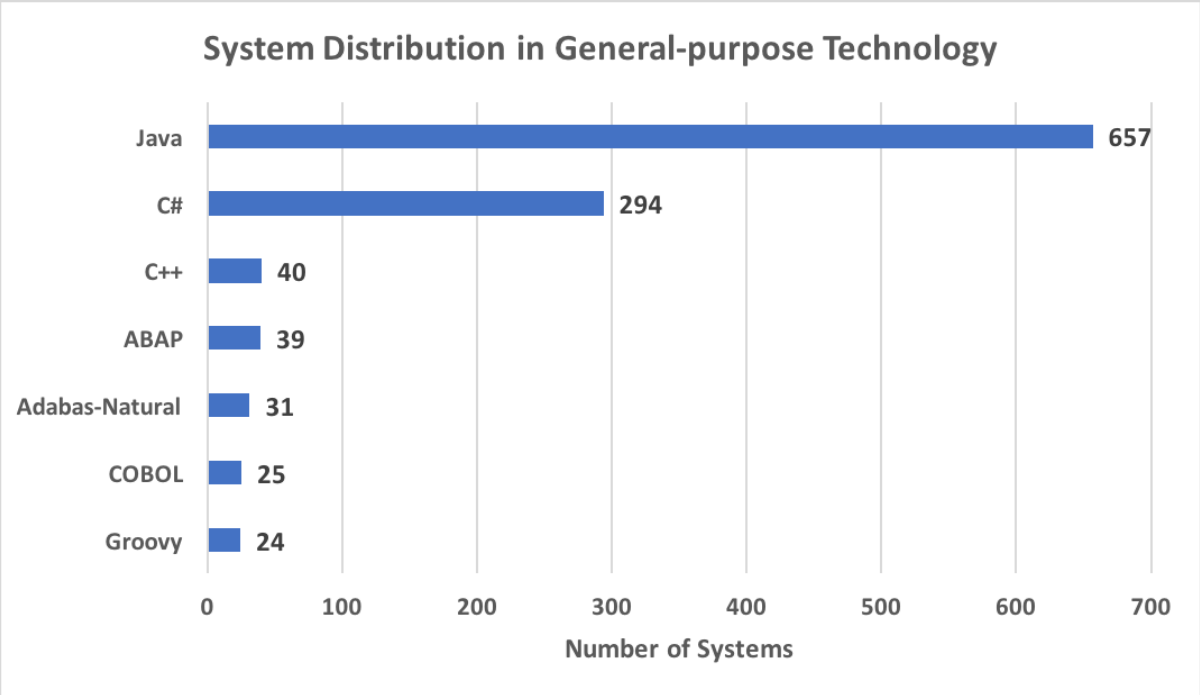
Groovy	It is an object-oriented programming language for the Java Platform. (Wikipedia, 2017)	It is useful as both a scripting language and also as a general-purpose language. (Quora, 2017)	(Groovy, GSP)
GSP	GSP is an acronym of Groovy Server Pages. It is a view technology which can be used designing web application using Grails Framework. (Srinivasan, 2017)	It is used to design web application.	(Groovy, GSP)
HTML	Hypertext Markup Language (HTML) is the standard markup language for creating web pages and web applications. (Wikipedia, 2017)	It is used for web applications.	(Java, HTML)
Java	Java is a general-purpose computer programming language that is concurrent, class-based, object-oriented, and specifically designed to have as few implementation dependencies as possible. (Wikipedia, 2017)	Java has a vast of different uses: - Website development - Networking - Data processing - Database connectivity - … (Srinivasan, 2017)	(Java) (Java, JSP) (Java, JavaScript) (Java, JSF) (Java, HTML) (Java, FreeMarker) (Java, XML) (Java, XSD) (Java, XML, XSD) (Java, XML Framework) (Java, PL/SQL) (Java, JavaScript, JSP, XML) (Java, JavaScript, JSP) (Java, FreeMarker, XML Framework) (Java, JavaScript, FreeMarker, XML)
JavaScript	It is a high-level, dynamic, untyped, interpreted run-time language. Alongside HTML and CSS, it is one of the three core technologies of World Wide Web content production. (Wikipedia, 2017)	JavaScript is well-suited for performing task within a web browser. It is primarily used to interpret with users. (Stack Overflow, 2017)	(Java, JavaScript) (Java, JavaScript, JSP) (Java, JavaScript, JSP, XML) (Java, JavaScript, FreeMarker, XML) (C#, ASPX, JavaScript)
JSF	JavaServer Faces (JSF) is a Java specification for building component-based user interfaces for web applications. (Wikipedia, 2012)	It is used for building user interfaces for web applications.	(Java, JSF)
JSP	JavaServer Pages (JSP) is a technology that helps software developers create dynamically generated web pages based on HTML, XML, or other document types. (Wikipedia, 2017)	It is helpful for creating dynamically generated web pages.	(Java, JSP) (Java, JavaScript, JSP) (Java, JavaScript, JSP, XML)
PL/SQL	Procedural Language/ Structured Query Language is Oracle Corporation's procedural extension for SQL and the Oracle relational database. (Wikipedia, 2008)	It is used to perform database operations. (Dummies, 2017)	(PL/ SQL) (Java, PL/SQL)
Razor	Razor is an ASP.NET programming syntax used to create dynamic web pages with the C# or Visual Basic .NET programming languages. (Wikipedia, 2017)	It is used to create dynamic web pages.	(C#, Razor)
T-SQL	Transact-SQL (T-SQL) is Microsoft's and Sybase's proprietary extension to the SQL (Structured Query Language) used to interact with relational databases. (Wikipedia, 2014)	It is used to interact with relational database.	(C#, ASPX, T-SQL) (C#, ASPX, T-SQL, XML)

XML	XML: Extensible Markup Language (XML) is a markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable. (Wikipedia, 2017)	XML is a technology for managing data exchange. It is a generic data storage format that comes bundled with a number of tools and technologies that should make it easier to exchange specific XML 'applications' between incompatible systems. (Wikibooks, 2017).	(Java, XML) (C#, XML) (Java, XML, XSD) (Java, JavaScript, JSP, XML) (Java, JavaScript, FreeMarker, XML) (C#, ASPX, T-SQL, XML) (Java, XML Framework) (Java, FreeMarker, XML Framework)
XSD	XSD is an acronym of XML Schema Definition. It specifies how to formally describe the elements in an Extensible Markup language (XML) document. It was designed with the intent that determination of a document's validity would produce a collection of information adhering to specific data types. (Wikipedia, 2017)	XSDs are documents that specify the structure of an XML document and help in their validation. (Stack Overflow, 2017)	(Java, XSD) (Java, XML, XSD)

Appendix C. System Distribution in Each Technology Functionality Type

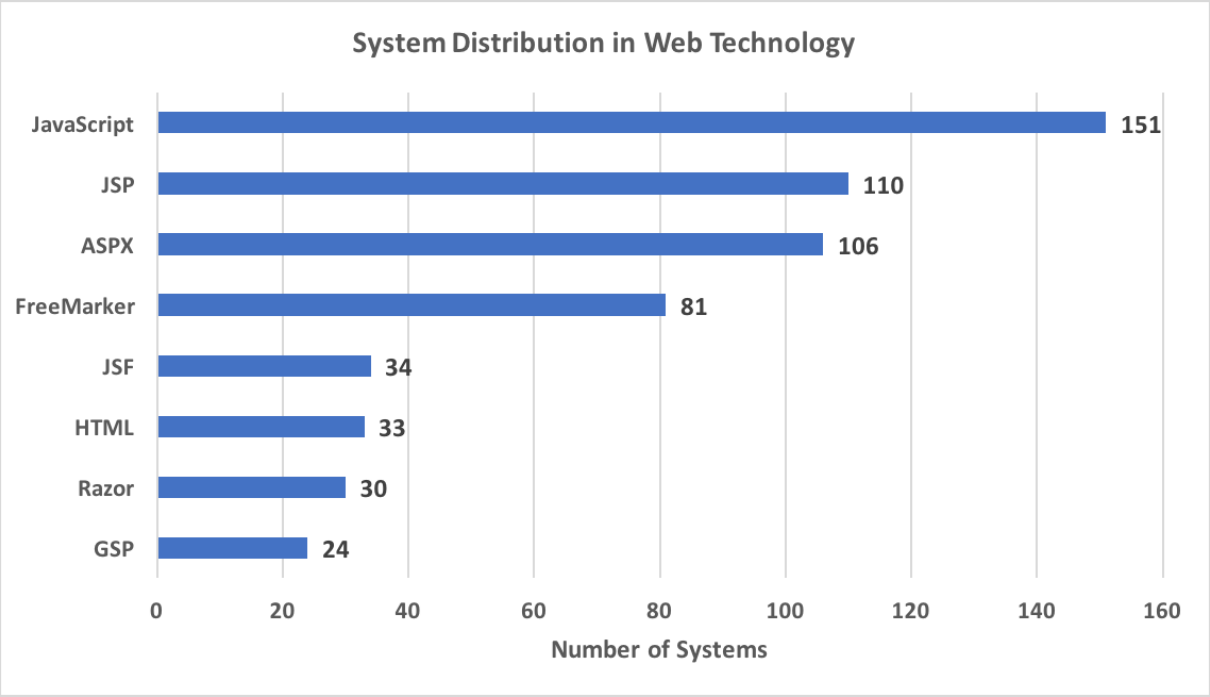
System Distribution in General-purpose Technology

General-purpose	Related Technology Stacks	Number of Systems	
Java	(Java)	161	Total: 657
	(Java, JavaScript)	36	
	(Java, JSP)	49	
	(Java, JSF)	34	
	(Java, HTML)	33	
	(Java, FreeMarker)	29	
	(Java, JavaScript, JSP)	38	
	(Java, PL/SQL)	36	
	(Java, XML)	93	
	(Java, XSD)	25	
	(Java, XML Framework)	23	
	(Java, XML, XSD)	25	
	(Java, FreeMarker, XML Framework)	26	
	(Java, JavaScript, FreeMarker, XML)	26	
	(Java, JavaScript, JSP, XML)	23	
C#	(C#)	133	Total: 294
	(C#, ASPX)	27	
	(C#, ASPX, JavaScript)	28	
	(C#, Razor)	30	
	(C#, XML)	25	
	(C#, ASPX, T-SQL)	25	
	(C#, ASPX, T-SQL, XML)	26	
ABAP	(ABAP)	39	Total: 39
Adabas-Natural	(Adabas-Natural)	31	Total: 31
C++	(C++)	40	Total: 40
COBOL	(COBOL)	25	Total: 25
Groovy	(Groovy, GSP)	24	Total: 24



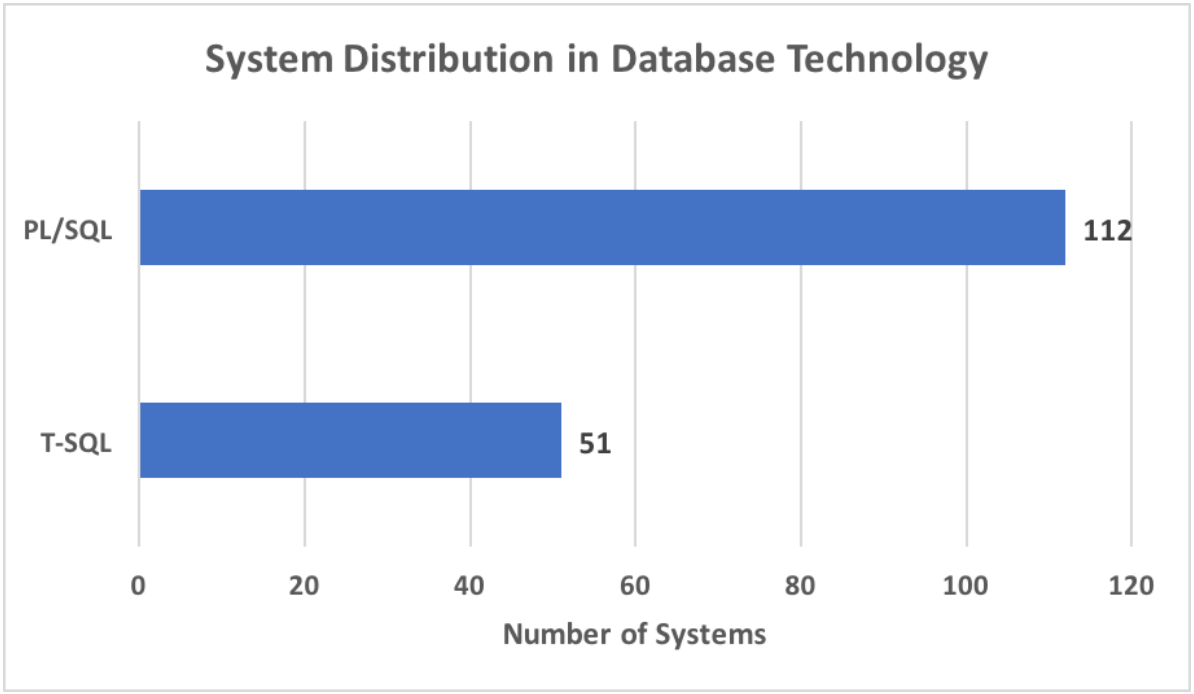
System Distribution in Web Technology

Web Technology	Related Technology Stacks	Number of Systems	
ASPX	(C#, ASPX)	27	Total: 106
	(C#, ASPX, JavaScript)	28	
	(C#, ASPX, T-SQL)	25	
	(C#, ASPX, T-SQL, XML)	26	
FreeMarker	(Java, FreeMarker)	29	Total: 81
	(Java, JavaScript, FreeMarker, XML)	26	
	(Java, FreeMarker, XML Framework)	26	
GSP	(Groovy, GSP)	24	Total: 24
HTML	(Java, HTML)	33	Total: 33
JavaScript	(Java, JavaScript)	36	Total: 151
	(Java, JavaScript, JSP)	38	
	(C#, ASPX, JavaScript)	28	
	(Java, JavaScript, FreeMarker, XML)	26	
	(Java, JavaScript, JSP, XML)	23	
JSF	(Java, JSF)	34	Total: 34
JSP	(Java, JSP)	49	Total: 110
	(Java, JavaScript, JSP)	38	
	(Java, JavaScript, JSP, XML)	23	
Razor	(C#, Razor)	30	Total: 30



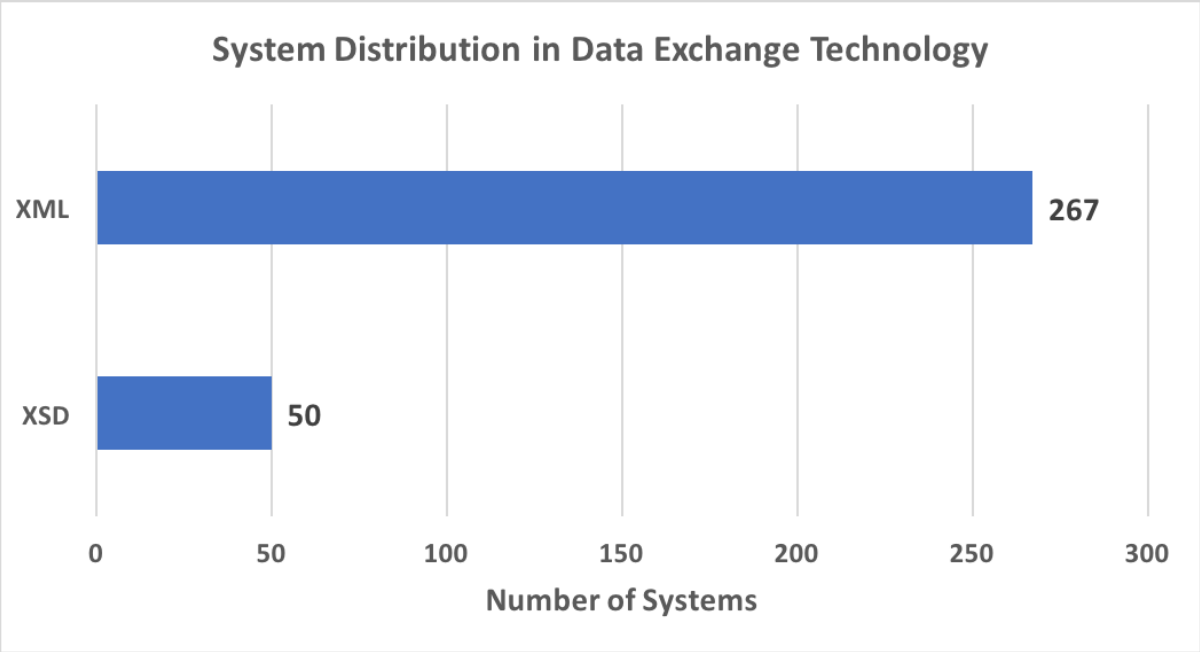
System Distribution in Database Technology

Database Technology	Related Technology Stack	Number of Systems	
PL/SQL	(PL/SQL)	76	Total: 112
	(Java, PL/SQL)	36	
T-SQL	(C#, ASPX, T-SQL)	25	Total: 51



System Distribution in Data Exchange Technology

Technology for Data Exchange	Related Technology Stack	Number of Systems	
XML	(Java, XML)	93	Total: 267
	(Java, XML, XSD)	25	
	(C#, XML)	25	
	(Java, JavaScript, FreeMarker, XML)	26	
	(Java, JavaScript, JSP, XML)	23	
	(C#, ASPX, T-SQL, XML)	26	
	(Java, XML Framework)	23	
	(Java, FreeMarker, XML Framework)	26	
XSD	(Java, XSD)	25	Total: 50
	(Java, XML, XSD)	25	



Appendix D. Technology Proportion in Industries

General-purpose Technology (Java & C#) Proportion in Industries

	Java	C#
Banking	81%	19%
Financial Services	50%	50%
Insurance	75%	25%
Energy	53%	47%
Technology-Software & Computer Services	55%	45%
Telecommunications	66%	34%
Industry Goods & Services-Transportation	70%	30%
Customer Goods	56%	44%
Health Care	36%	64%
Government	68%	32%
Average	**61%**	**39%**

Web Technology Proportion in Industries

	ASPX	Razor	GSP	HTML	FreeMarker	JSF	JSP	JavaScript
Banking	6%	3%	0%	1%	40%	0%	17%	33%
Financial Services	22%	11%	0%	0%	0%	22%	17%	28%
Insurance	16%	13%	0%	3%	0%	6%	36%	26%
Energy	19%	12%	0%	6%	6%	0%	38%	19%
Technology-Software & Computer Services	53%	4%	0%	6%	0%	7%	12%	18%
Telecommunications	21%	16%	10%	21%	5%	0%	11%	16%
Industry Goods & Services-Transportation	26%	2%	0%	4%	0%	13%	20%	35%
Customer Goods	28%	0%	0%	0%	0%	0%	43%	29%
Health Care	50%	0%	0%	0%	0%	0%	25%	25%
Government	13%	6%	18%	14%	0%	10%	21%	18%
Average	**25%**	**7%**	**3%**	**5%**	**5%**	**6%**	**24%**	**25%**

Database Technology Proportion in Industries

	PL/SQL	T-SQL
Banking	60%	40%
Financial Services	88%	12%
Insurance	83%	17%
Energy	80%	20%
Technology-Software & Computer Services	16%	84%
Telecommunications	100%	0%
Industry Goods & Services-Transportation	75%	25%
Customer Goods	100%	0%
Health Care	67%	33%
Government	90%	10%
Average	**76%**	**24%**

Data Exchange Technology Proportion in Industries

	XML	XSD
Banking	88%	12%
Financial Services	85%	15%
Insurance	69%	31%
Energy	90%	10%
Technology-Software & Computer Services	89%	11%
Telecommunications	69%	11%
Industry Goods & Services-Transportation	58%	31%
Customer Goods	100%	0%
Health Care	100%	0%
Government	74%	26%
Average	**85%**	**15%**

Appendix E. Business Application Classification Benchmark

Business Application Classification Benchmark

	System Group		System Type	
Primary Systems	110	Design Engineering & Development	111	Computer Aided Design
			112	Computer Aided Manufacturing
			113	Computer Aided Engineering
	120	Analytical Applications	121	Algorithmic Systems
			122	Statistical Systems
			123	Decision Support Systems
	210	Process Controllers	211	Creational Process Controller
			212	Non-creational Process Controller
	220	Transaction Processing Systems	221	Batch Transaction Processing Systems
			222	Real-time Transcation Processing Systems
	230	Resource Management System	231	ERP
			232	Managed Resource Planning
			233	Inventory Control
			234	Resource Allocation
			235	Supply Chain Management
	240	Case or Event management	241	Incident Management System
			242	Auditing Support System
	250	Interfacing Systems	251	Client Portal
			252	Identity & Access Management
Supportive Systems	310	Communication	311	Asynchronous Communication
			312	Synchronous Communication
	320	Functional Applications	321	Human resource Application
			322	Financial Billing System
			323	Sales/ Customer Relationship Management
			324	Legal Application
			325	Facility Management
			326	Management Information System
			327	Marketing
	330	Knowledge and Document Management	331	Knowledge Management
			332	Document Management
			333	Content Management System
	340	Personal Productivity	341	Office Productivity
			342	Note Taking Appliction