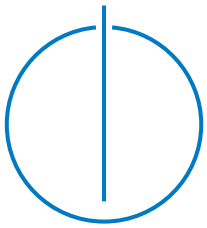
Fakultät für Informatik
Der Technischen Universität München

Master's Thesis in Informatik

User-Adaptable Rule-Based Natural Language Generation for Regression Testing.

Anupama Sajwan



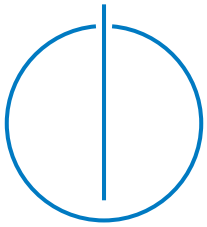
Fakultät für Informatik
Der Technischen Universität München

Master's Thesis in Informatik

User-Adaptable Rule-Based Natural Language Generation for Regression Testing.

Benutzergesteuerte regelbasierte Generierung natürlicher Sprache für Regressionstests.

Author: Anupama Sajwan
Supervisor: Dr. Prof. Florian Matthes
Advisor: M.Sc. Daniel Braun
Date: January 15, 2018



Declaration

I assure the single handed composition of this master's thesis only supported by declared resources.

Munich, January 15, 2018

Anupama Sajwan

Acknowledgement

I would first like to express my sincere gratitude towards *Prof. Dr. Florian Matthes* for giving me an opportunity to pursue this research. At the initial stage of my thesis, his hard questions and encouragement widen my understanding and research's perspective.
I would like to thank my advisor *Daniel Braun* for constantly supporting me and steering me in right direction. He always fueled my curiosity with his insightful suggestions. Thank you for being patient with me throughout the research.
I would also like to thank the experts who were involved in the complete research cycle. They helped me to understand the usecase and expectations they have from this research. They not only motivated me in hard times but also remained transparent regarding their feedback.
Last but not least, I would like to thank my *Friends & Family* for supporting me in tough time and helping me with every way possible.

Abstract

In today's environment, domain experts are flooded with data, as their job is to analyze data and provide feedback to peers or other people. Their task is too laborious and time-consuming, therefore they need a software which can help them with their work by automating it.
The focus of this thesis is to carry out a research on current state of art in field of "Natural Language Generation" (NLG). In addition, implement a pilot project which can automate the process of data analysis and expresses the result in form of textual reports.
Case under investigation is related to the financial domain, hence the developed tool's logic should be transparent to business users; it should not be a black box. To increase the effectiveness of the tool, it should be user-adaptable.
In the end, the developed tool is evaluated to check its effectiveness and user acceptance.

Contents

List of Figures V

List of Tables VI

Abbreviations VII

1. Introduction 1

1.1. Motivation and Objective 1
1.2. Background 3
1.2.1. Insurance Companies 3
1.2.2. Solvency II 3
1.2.3. Regression Testing Use Case 5
1.2.4. Proposed Methodology 6
1.3. Research Questions 7

2. Related Work 8

2.1. Scientific Work 8
2.1.1. FOG (Forecast Generator) 9
2.1.2. SumTime-Turbine 9
2.1.3. BabyTalk (Automatic generation of textual summaries from neonatal intensive care unit) 10
2.2. Commercial Tools 11
2.2.1. YSEOP 11
2.2.2. AX Semantics 13
2.3. Rules Engine 15
2.3.1. DROOLS Engine 16
2.3.2. Camunda DMN 16
2.4. Natural Language Generator 17

3. Requirements Analysis **20**
3.1. Requirements Specifications 20
3.1.1. Input Attributes 20
3.1.2. Expected Output 21
3.1.3. Functional Requirements 22
3.1.4. Non-Functional Requirements 25
3.2. Analysis 26
3.2.1. Rules Engine 26
3.2.2. Natural Language Generator 27
3.2.3. System Model 28
3.2.4. Static System Model 28
3.2.5. Dynamic System Model 29

4. Design and Implementation **31**
4.1. Design 31
4.1.1. Architecture 31
4.1.2. Business Process 32
4.2. Implementation 33
4.2.1. Development Environment 33
4.2.2. Software Packages 35
4.2.3. User Interface Navigation Flow 37

5. Evaluation **42**
5.1. Methodology 42
5.2. Evaluation Results 44

6. Conclusion **47**

7. Future Work **49**

Appendix A. Evaluation Questionnaire for Experts **i**

Appendix B. Evaluation Questionnaire (Online) for General People **v**

Appendix C. AX-Semantics Screenshots **x**

Bibliography **xiv**

List of Figures

1.1. Basic Data Driven Decision System 2
1.2. Basic Workflow of Project 5
1.3. Proposed Methodology 7

2.1. Basic Workflow of YSEOP's Solution 12

3.1. Use Case Diagram 24
3.2. Static Model: Component Diagram 29
3.3. Dynamic Model: Activity Diagram 30

4.1. Business Process 33
4.2. Package Diagram 35
4.3. Screen 1: Home Screen 37
4.4. Screen 2: Start Process 38
4.5. Screen 3: Choose File 38
4.6. Screen 4: Select Rules 39
4.7. Screen 5: Show Generated Text 39
4.8. Screen 6: Downloaded Text File 40
4.9. Sequence Diagram 41

5.1. Survey: Age and Gender 44
5.2. Survey: Questions about Text 45
5.3. Survey: Questions about NLG Tool 46

C.1. Create Project x
C.2. Basic Steps x
C.3. Create Collection xi
C.4. CSV File with Data xi
C.5. Upload Collection Data xii
C.6. Create Text Containers xii
C.7. Generate Phrase Containers xii
C.8. Results xiii

List of Tables

3.1. NLG requirement 22
3.2. NLG quantifiers requirement 23
3.3. NLG requirement for scenario change 23

4.1. Development environment of NLG tool 34

Abbreviations

AI	Artificial Intelligence
API	Application Program Interface
BPMN	Business Process Model and Notation
BRMS	Business Rules Management System
DDD	Data Driven Decision
DMN	Decision Model and Notation
NL	Natural Language
NLG	Natural Language Generation
NLP	Natural Language Processing
SCR	Solvency Capital Requirement
UML	Unified Modeling Language
VaR	Value at Risk

1. Introduction

In this chapter, motivation of the thesis is described and its objective is established.

1.1. Motivation and Objective

In last 3 decades, Artificial Intelligence (AI) has advanced to such an extent that it has the potential to predict the future. These predictions has empowered human to take better decisions to minimize risks and plan their tasks in an optimized way [BS]. These data-driven decision making [PF13] usually consists of 4 steps [Rob] [LF] [Br] shown in Figure 1.1:

- Process/System Modeling: At first, aim and requirements of the Data Driven Decision (DDD) System are listed, then the process or system is modeled.
- Collect Data: Then "data" for analysis is collected. In case under investigation, data is generated by computing machines.
- Data Interpretation: Then the generated data is analyzed by domain experts to gather insights.
- Recommendation and Feedback: According to the analysis of the experts, feedback are provided. Based on the experts recommendations, further decisions are taken or DDD processes are modified.

This DDD approach is followed in wide range of industries including manufacturing, finance, marketing, customer care and so on [Bu]. Industries which are driven by "predicted data" need to run these analyses more frequently. Due to this demand, experts are often flooded with data and start competing with machines [UF96]. Therefore, there is a need for a tool which can automate this process and helps experts in their daily analysis job.

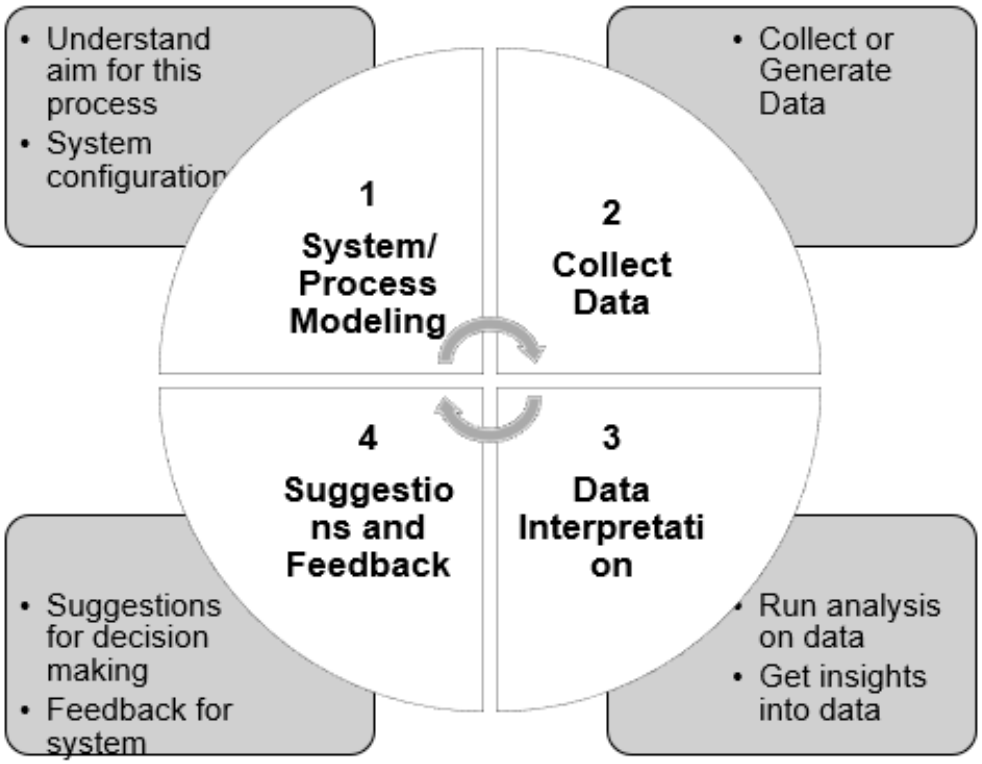


Figure 1.1.: Basic Data Driven Decision System

The usecase under investigation is "Regression Testing Results of Solvency II Risk Capital Calculation". Regression testing on various instruments and scenarios based on Solvency II, results in a large amount of data. This data is collected in spreadsheets in form of figures (i.e. numbers) under various risk categories, which is analyzed by experts to provide reports/feedback. As these analyses directly influence the funds of the company, more frequent or quicker analyses will be profitable and favorable. In the near future, as the frequency of test runs will be increased to a weekly mode, even more data needs to be scrutinized by experts in the same time frame.

Therefore, the goal of this thesis is to implement a pilot project to improve productivity of the experts. The idea is to formulate rules to analyze data and represent the results in form of text by using Natural Language Generation (NLG) technique. "NLG is a subfield of Natural Language Processing (NLP), which generates language in form of text or speech from a machine representation system such as a knowledge base or a logical form" [Wib]. This thesis includes analysis of two commercial tool (YSEOP) and AX-Semantics; and to compare this tool with a shortlisted open source NLG plugin; the pilot is

to be implemented using an open source NLG engine. As a result, experts would have an automated tool set which will support them in their day to day analysis. In addition, this thesis is also targeted towards reducing the cost and optimize the benefit of NLG solution. So, it is aimed to create a solution, which can be modified by domain experts and is therefore less binded to specific usecase.

1.2. Background

1.2.1. Insurance Companies

Insurance companies main business is to insure people, property or companies. They provides financial protection to the insurers against loses, in return they collect a monthly premium amount from them [Ina]. Therefore, they earn money from the premium paid by the insurers, which is further utilized to invest in market. As they also has liabilities like dividend of shareholders or claims of insurers [Roa], there is a need to strike a balance between money to be invested and money to be saved. Here comes the role of Solvency II.

1.2.2. Solvency II

Solvency II is the set of rules and regulations that must be followed by EU insurance companies. It enforces them to analyze their risk and safeguard minimum risk capital to reduce the risk of insolvency [Wie]. Hence, it bolsters the financial stability of these companies and provide confidence to the market. Its main goal is "to protect policy holder's rights, so that insurance companies are able to pay insurance claims".
Solvency II has three pillars approach [Tu07] [Ei15]:

First Pillar: Quantitative Capital Requirement A market consistent framework is used by insurance firm to calculate their "available capital" by economic evaluation of the assets it acquires and liabilities it is obliged to pay. Then "Solvency Capital Requirement" (SCR) is calculated using either Standard Risk Model or Internal Risk Model.

Second Pillar: Supervisory review of risk management The analysis of strengths and weaknesses of risk management techniques is done along with documentation of system and controls. According to the analysis, risk management techniques are adopted.

Third Pillar: Transparency and disclosure requirements As insurance firms should be transparent regarding their risk strategy, they have to disclose "Solvency and Financial Condition" (SFC) report to the public. This report includes capital information; risks related figures; and methodology to calculate and manage risks.

Risk capital can be estimated with the help of two different techniques:

Standard Risk Model : It is the general risk model provided by Solvency II regulators which an insurance firm utilizes to calculate the "risk capital requirement". As the *Standard Risk Model* is based on conservative approach, it estimates a huge amount of risk capital, as a result, lesser capital is available for investments and expenses. Usually, the standard model is used by smaller insurance firms that have limited resources to invest in Solvency II calculations.

Internal Risk Model : As the name suggests, this risk model is developed internally by an insurance firm, which is specific and customized to its operations and processes. As it is designed specifically for one firm, so it has intricate information about the risks, processes, investments of it. Therefore, it is optimized to calculate lower risk capital requirements [Gi08], which helps the organization to save a small percentage of their earnings. This minor percentage of earnings can be equivalent to a huge amount of money for bigger firms than smaller firms. This saved money can be used for further investment and growth of the firm. But the internal model must be approved by the regulators first, which requires detailed documentation of the risk and scenarios to justify the assumptions and methodologies employed to build the internal model. So, the trade-off in this approach is "to invest time and human resources to create an internal model" versus "the amount of money gained by saving margin in risk capital".

1.2.3. Regression Testing Use Case

In this section, the use case under investigation is described.

As previously discussed, insurance companies collect money from policy holders and invest it to gain profit. With the profit, they pay the liabilities or obligations of their insurance contracts. Their nature of business has a higher risk as their bankruptcy would impact both organization and policy holders. So, to ensure the protection of policy holders all European countries's insurance companies are enforced to follow the Solvency II directive.
The use case under investigation is "Regression Testing for Solvency II Risk Calculation". This process is an integral part of the risk management system of the company, shown in Figure 1.2. Risk Management is the process of assessing, managing and mitigating losses, which reduces potential of insolvency. It consists of a sophisticated internal risk model which uses a Monte Carlo simulation to predict the risk values of all the investments and assets of the company [VP14]. These generated figures are utilized by experts for analysis of risks embedded in the business of company. Then the experts forward the result of the analysis to higher management. The result of the analysis gives an insight into the risk associated with the company's activities. It can also assist Board members to take strategic decisions regarding further business expansion or acquisition or supervision of their internal processes.

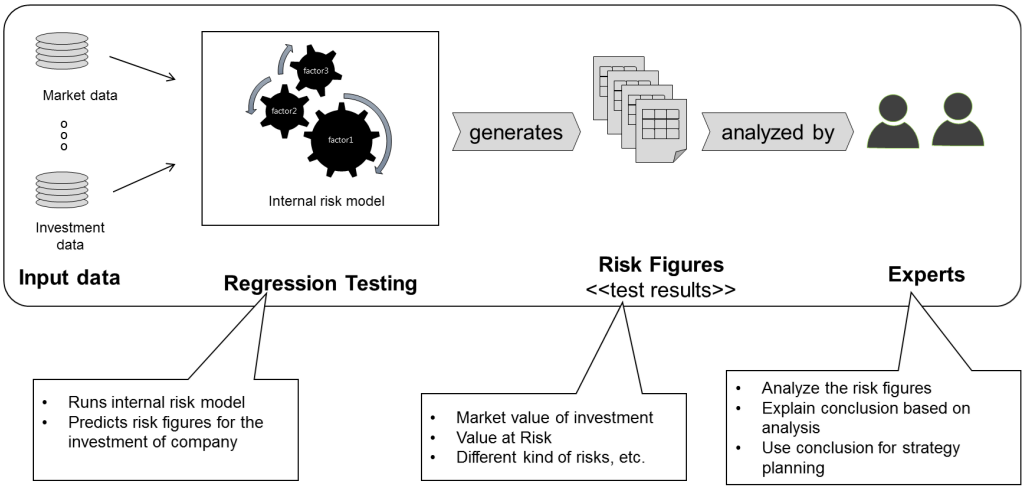


Figure 1.2.: Basic Workflow of Project

It is called Regression Testing because experts compare risk figures of two consecutive runs. The first run is called "Base Run" and the second run is

called "Regression Run". In between Base Run and Regression Run some changes are introduced either in the internal model code or in the input data of the model. Experts usually do not expect any difference between the risk figures of two runs. So, the reason for a change of values could be:

1. Change in the internal model code is not appropriate, it has some errors. In this case a bug is reported to the software development team.
2. Experts introduced some changes through input values, but output values are not same as expected, in this case deep analysis of risk figures is required.
3. Changes are introduced by the changing economy or some risk factors which are in synchronization with the market values or other factors. This also requires deep analysis, as it can affect business of the insurance companies.

Therefore, in this process experts need to analyze data at regular intervals for following reasons:

1. Maintaining the quality of the internal model software with changing business requirement.
2. Stay alert for changing market and other risks, to minimize loss and maximize profit of organization.
3. To provide inputs to higher management, which can assist in improved risk understanding and further helps in strategic and organizational planning.

1.2.4. Proposed Methodology

As analysis of risk figures is very laborious and consumes enormous amount of time on the side of experts, so this process needs to be automated. To automate this analysis, one needs to incorporate the knowledge of experts into the software and enable it to express the analysis results in natural language, as shown in Figure 1.3. Therefore, the resulting software contains two parts:

1. Rules Engine: It incorporates the knowledge of experts in the form of business rules.

2. Natural Language Generation (NLG): It has the ability to express the result of analysis in form of natural language, that is, text.

The resulting software or pilot project will help experts in their analysis and will speed up the process of analysis.

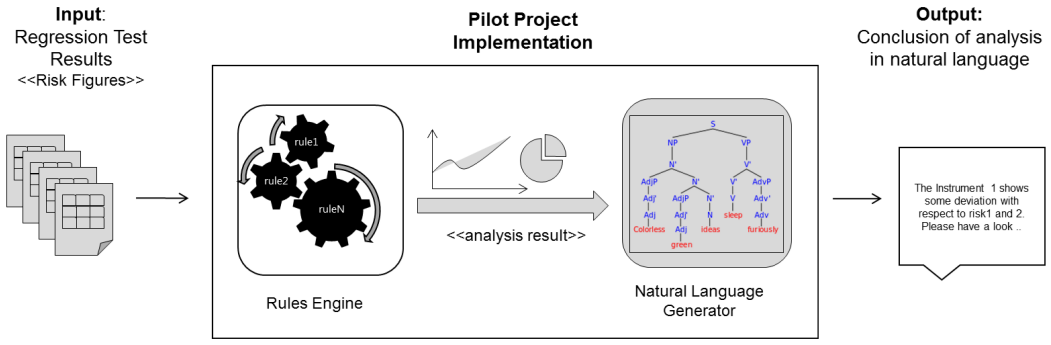


Figure 1.3.: Proposed Methodology

1.3. Research Questions

The following research questions will be answered by the thesis:

1. How can expert knowledge be converted into user-adaptable computation rules?
2. How can textual reports be created based on the results of these rules?
3. How can such a system be implemented using open-source plugins or tools?
4. How can a NLG system be made user-adaptable and loosely-bound to a specific use case?
5. Can machine generated textual reports improve the operational excellence of employees?

2. Related Work

The idea behind the thesis and the pilot project is to extract expert knowledge in machine readable format and use it for analyzing data. Afterwards NLG will be used to explain the result of analysis. This chapter summarizes the related research carried out with respect to NLG. It also provides an analysis of current commercial NLG solutions.

2.1. Scientific Work

In 1996, "The KDD Process for Extracting Useful Knowledge from Volumes of Data" paper [UF96] was published, which completely agree that with advancement of Computer Science and availability of inexpensive hardware storage, volume of stored data has increased. This data is usually utilized for extracting knowledge or patterns and creating reports. These report can be utilized for benefits of institutes, organizations or people. But volume of data is increasing exponentially and capacity of human to analyze it and extract information is limited. So, there is a need of automating the process of information extraction.

As the aim of this thesis is to assist experts in data driven decision making, by extracting information from data. Therefore, there is a need to understand what kind of representation of information helps human to take better decisions. In this regard, one research was conducted by Edinburgh Napier University and Heriot-Watt, where they developed a game on human decision making [LR16]. In this game, weather information is presented sometimes in the form of text; or sometimes in the form of chart; or sometimes a combination of both chart and text. Users were exposed to this weather information and they had to understand it and predict weather. They also had to note their confidence level that their prediction is correct. For every correct prediction they were given some points. After the analysis, the research shows

that although users were not confident regarding text information, but they earned the highest points. In case of decision making, this research highly supports natural language (text) as a medium to convey information. Similarly, in our use case experts also need to analyze data and make decisions, therefore natural language output would be more suitable for this pilot project implementation. Hence, further research is focused on NLG.
In past, there has been many research carried out in area of "automatic analysis of data and producing textual summary". Following sections contains some research focused on this area.

2.1.1. FOG (Forecast Generator)

In 1994, the application FOG [EG94] was developed to analyze time-series weather data and generate textual summaries. It has three stages: concept formation, text planner and text realizer. *Concept formation* is the first stage in which time series data is extracted from a predicted weather chart. Then with the help of rules, data which is relevant and important according to different variables like wind speed directions is shortlisted. Then from the shortlisted data "concepts" are formed, which are fed into the second stage *text planning*. During *text planning* these unstructured concepts are filled with some abstract information and further organized into sentences sequences. In the last stage of *text realization*, this information is filled with a lexicon and corrected by grammatical rules, so this stage has a language-specific processing so this stage is according to the language of output text. In this project, they fed the last stage with two languages in parallel, English and French, so they could produce textual reports simultaneously in two languages.

2.1.2. SumTime-Turbine

In 2004, Sumtime [Yu04] prototype was developed to analyze gas-turbines sensors data. The aim of the prototype was to predict non-favorable events and anomaly detection. Ultimate goal was to help in optimizing availability of turbines and reducing the maintenance cost. Its architecture consists of advanced data analysis techniques like: pattern recognition, pattern abstraction and pattern selection; and text generation which includes: text-planning and

realization. In *advanced data analysis*, at first data is analyzed to extract patterns, followed by pattern abstraction, then significant patterns are selected. These selected patterns are used to plan text-structure. Finally, based on the text plan, a textual summary is generated. Later, the Sumtime prototype was implemented in SumTime-Turbine system, which generated textual summaries of gas-turbines time-series data.

2.1.3. BabyTalk (Automatic generation of textual summaries from neonatal intensive care unit)

In 2008, BabyTalk [Po08] was a project developed in collaboration of Neonatal Intensive Care Unit, University of Aberdeen and Clevermed Ltd. This project is aimed to analyze data produced in Neonatal Care Units and generate textual summaries for different audience. It combines intelligent signal processing and NLG. As input it has three kinds of data: time series data from physiological sensors, structured information about events and free-noted from the medical staff. At first it analyzes time series signal data and then with the help of events it interprets data. Then based on results of the data interpretation, it plans the document structure which include forming abstract structure of sentences and sequencing them. Then according to document structures it finally realizes sentences by filling them with language-specific lexicon and structuring with language-specific grammatical rules. It had many errors and problems like *continuity problem*, where different parts of text are not consistent with each other; it was not able to differentiate the cause of signal change between ˝sensor problem˝ and ˝real physiological change˝; and poor long term overview. But it successfully established that NLG along with Artificial Intelligence techniques can aid decision-making.

All the above discussed projects are a few of the early projects in NLG, which focus on *data-to-text* approach. Their main goal was to help domain experts with their daily data analysis work. But NLG systems implementation is challenging and has concerns like: setup of knowledge base. At the initial stage of a NLG system creation the following challenges are faced [RM93]:

- Use case does not have an infrastructure or database which can be utilized for language generation.

- If they have infrastructure, it is not suitable for a NLG system.

Some researchers also tried to help for overcoming these challenges by inventing tools for example:

- Linguistic assistant for Domain Analysis (LIDA) [OR01] helps domain experts to bridge gap between domain knowledge and technical specifications. It helps to translate user's knowledge to UML diagrams.
- A tool developed for automatic customization of a commercial natural language system according to a sales and marketing database [WNS92]. Here, knowledge of database and Entity Relationship (ER) diagrams was translated into Natural Language (NL) transcripts, which were run on natural language system for customization.

Usually NLG systems are very tightly bound to the context or use case, therefore there is still need for NLG systems which are user-adaptable or can be utilized for more than one use case [Co97].

2.2. Commercial Tools

There are several commercial tools which claim that they are equipped with the latest technology of NLG. Out of these commercial tools, two tools selected for evaluation are YSEOP and AX-Semantics. One of these tools which holds a big share of the NLG market is YSEOP [EE] [YS]. The following contains the detailed findings from the interaction with YSEOP company representatives.

2.2.1. YSEOP

YSEOP is a global enterprise and one of the leading companies which provides commercial NLG solutions [EE] [YS]. The underlying technologies for its solution are: Java, Tomcat and YML (YSEOP Markup Language). YML is a XML-based language, developed by YSEOP, it is used to formalize rules of analysis. The knowledge of the user is coded in the form of Java classes and rules. For example, a "Fund" is a Java class and it has different variables like interest rate, equity rate etc. The "data-to-text" workflow starts with providing data in form of excels sheet. As YSEOP's solution can only access

data in YAML format only, so these excel sheets are converted in YAML format. Then, generated YAMLs are fed to a rules engine which analyses the data. The output of rules engine helps with the selection of templates and formalizing the text. In the end, the final text is produced and sent back to user.
YSEOP provides two kinds of solutions:

Webservice API : It provides a web application to formalize rules and to create templates for text generation. The user sets conditions when a sentence should appear or when it should not appear. In addition, the user can provide more than one template for the same sentence; this helps to a create variety of reports. Once rules and templates are set up in the server, the user can send data to server through webservice calls and subsequent text is generated as a response. These webservice calls are in XML format, containing data which needs to be analyzed by rules. The user has the flexibility to change rules or text fragments.

Standalone Java Project : It is a Java project, which contains the codebase of NLG. It is stored locally on the computer and it can be accessed with the help of the Eclipse IDE. Here the user can also create or change rules and text fragments, but it is not user-friendly. Only if the user has knowledge of Java language, it can be used.

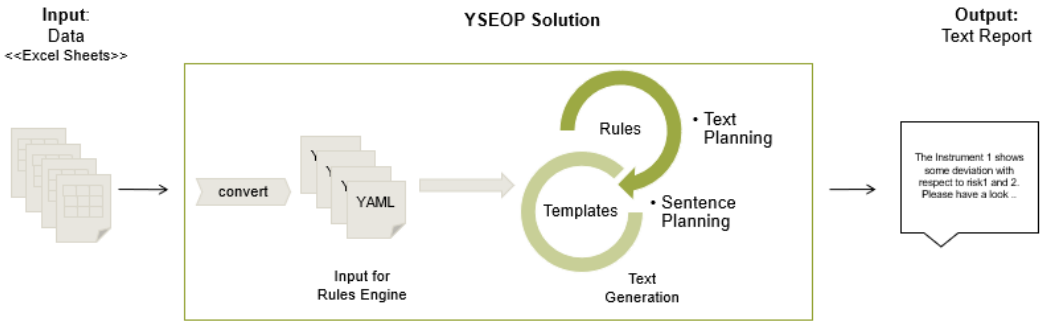


Figure 2.1.: Basic Workflow of YSEOP's Solution

After analysis of the YSEOP's solution, following advantages and disadvantages have been identified. Advantages of their solution are following:

- Multilingual text: As they create and store templates for all target languages, the output of the rule engine can be processed by different lan-

guage templates separately and in parallel. Therefore, multilingual text is generated at the same speed as single language text.

- Transparency to users: As the user can see which rules are leading to the output text, they trust the Language Generation system.
- Flexibility to users: As it provides an easy User Interface to business people to create or modify rules and text templates, the business user can work without the need of IT support.

Disadvantages of their solution:

- No data analysis: Their solution does not support data analysis. Only possible way to incorporate data analysis is by analyzing data prior to feeding it in the solution.
- Vendor Lock-in: As YSEOP uses a specific language (YAML) to encode business rules, in the future, if users want to switch vendor, business rules need to be formulated again from scratch.
- Webservices data security: As a ˝standalone solution˝ needs java knowledge, a ˝webservices solution˝ is favorable. But in a productive scenario, for every request, the user will send some data to the server, so there is always a risk of losing business information. If the server is installed on-premises, then there is a need of a DevOps team and Java developers to maintain the solution according to changing business needs.
- It consumes a lot of time and money to train users or experts. For setting up the solution for a new language a significant amount of time and money is required.

2.2.2. AX Semantics

Ax-Semantics is a company which provides SAS-based NLG Cloud solutions in 24 languages. It has free demo (sandbox) available online, which was evaluated for this research.
At first, the user creates an account on their official website[1], then the user

[1] AX-Semantics sing-up link: https://www.ax-semantics.com/de/sign-up.html

has to follow a basic workflow, as shown in Figure C.2. The basic workflow has the following steps:

1. **Data Source Creation** In this step, the user creates collections or tables and imports data. Data can either be pasted on the clipboard or can be uploaded via a data file. Data should be in excel or JSON or CSV format. For testing, "Books" collection is created and csv file with data, as shown in Figure C.4, is uploaded. This data is collected from Wikipedia [Wia].

2. **Compose Sentences** Here, user composes the template for expected output. It has 4 modes:

   - **Write Mode**: To create or modify the template text.
   - **Magic Mode**: To select the parts of sentences which should be filled from data. For example, in sentence: "ABC book is written by XYZ.", ABC should be replaced by books's name and XYZ should be replaced with author's name. Therefore, user can generate *Phrase Container* for them and link them to the respective column of the collection, as shown in Figure C.7.
   - **Variant Mode**: It allows to create different variation of the same sentence.
   - **Translate Mode**: This is to create equivalent sentences in different languages. The user can create different templates of same text in different languages.

3. **Generate Result** In the end, the user selects rows present in the collection, for which the user wants to generate text. For every row in the collection, one copy of the template is generated (with respective column's values), as shown in Figure C.8.

After analysis of AX-Semantics tool, it advantages are listed below:

- As this tool is SAS based, it is easy to understand for non-programmers. If data (database or collections) is ready according to the NLG needs, then non-programmers might not need technical help for simple sentence generation.

- This service is highly useful for repetitive and simple text generation for large datasets.
- It allows to create pairs between languages, which is also a useful feature for multilingual requirements.

Potential disadvantages of this tool are the following:

- As it is template-based NLG, it does not have grammatical checks after the sentence is generated.
- Although it enables the user to set up a complete NLG project by herself, but still it has a learning curve. It is not easy to use this tool without some prior technical knowledge.
- The user can only import data but there are no functions to analyze it. Therefore, data analysis is not possible and data needs to be analyzed before feeding it to the tool.

2.3. Rules Engine

As a NLG system needs to be feed with knowledge of experts, there is a need of a Rules component. The idea is to formalize business rules and encode them in a Business Rules Management System (BRMS). This BRMS should have the following capabilities:

1. Capable of expressing business rules.
2. Easy and simple for business users to understand.

Usually business users have a good knowledge of business processes and rules, but they face challenges to understand or translate them into technical code. DMN (Decision Model and Notation) [MS] and BPMN (Business Process Model and Notation) [Gr] are the open standards developed by *Object Management Group*. They fill the gap between business users and encoding of business processes and rules. Benefits of using BPMN and DMN standards are following [Ti17]:

- As they are widely known standards, these notations are in practice and understood by many business and IT people.

- As business knowledge is encapsulated in these standards, it is easier to switch to a new vendor.
- They allow to create a coherent model of the process and decisions [Ta].

For the *Regression Testing* use case, a rules engine which follows DMN standard should be followed. Currently, *Camunda DMN* and *Drools* claims to follow DMN standards. Therefore, these open source tools were compared based on previous research [BD16].

2.3.1. DROOLS Engine

Red Hat company provides jBPM and Drools Engine, which are free to use. But if customer support is needed, the user has to pay some fees. From 2017, they follow BPMN and DMN standards. It is based on Java platform and supports Tomcat, WildFly and JBoss servers. It has many features like flexible process writing, visual configuration of dashboard, filtering and search in memory [Or]. As it is an old and developed tool, the learning curve is very high for beginners. Lack of proper documentation also increases the difficulty for a new programmer. It is not scalable and cannot be used in clusters for a large database.

2.3.2. Camunda DMN

Camunda is a Java-platform based product of a Berlin-based company. It follows BPMN 2.0, DMN 1.1 and CMMN 1.1 standards. It has 2 types of licenses: *Community Edition* which is free and *Enterprise Edition* which has a fee and in return provides some extra features [Caa]. The *Enterprise Edition* has some additional features like: (i) business rules can be changed on the fly, no need to build and deploy the project for rule changes, (ii) Heatmaps of the process states are created, and (iii) simple management of history and processes data and measure KPIs [Caa]. Camunda for execution does not have any dedicated server or database. It supports a wide range of servers like: Apache Tomcat, JBoss AS, IBM WebSphere, Oracle WebLogic and WildFly Servers. Regarding database it can be integrated with MySQL, MariaDB, Oracle, IBM DB2, PostgreSQL, Microsoft SQL Server and H2 [Cab].

Camunda needs technical support, but it has a nicer structured design, which makes it user-friendly for business users. "Additionally the system can be used in a cluster, it is scalable and supports a multi-client capability" [BD16]. Therefore, its simplicity and scalability qualify it as rules engine for this pilot implementation. Also, the users of these use case are familiar with this tool and already have some solutions running on Camunda business processes. Hence, in the future this pilot project can be pipelined with the existing Camunda processes.

2.4. Natural Language Generator

In general, there are 3 types of text generation techniques:

1. **Template-based**: Template-based natural language generators are tools that use predefined sentences and blanks to be filled by the tool. They do not have knowledge of a language like: lexicon, grammatical rules, etc. According to Deemter (2003) [DKT03], these generators are brittle, difficult to maintain and less expressive. They just have pre-structured sentences which have few blanks to be filled by the tool. For example, in online shopping website the description of a shirt's template would be: "The <shirt name> is made from <material name> material." So, an output sentence of NLG would be: "The XXX shirt is made from 100% cotton." But if the shirt is made from cotton and polyester then it cannot change to: "The XXX shirt is made from 80% cotton and 20% polyester."

   AX-Semantics and YSEOP tools defined in previous section 2.2 also fall under this category. One more example for these types of tools is Exemplar [WC98] from CoGenTex Inc. It uses a combination of Java rules and pre-defined templates to generate text. After knowing the limitations of these systems, they are slowly evolved into systems which have a grammatical tree-structure for every sentence. One example of the tool which represents this phase of metamorphosis is TAG [Be02]. Here, the approach is to derive tree structures from templates and to try to add some grammatical context to the tool.

2. **Realizer-based**:Realizer-based NLG systems are based on *Realization* concept of linguistics. Realization is a process that converts abstract information or data to natural language [Wid]. Here, systems have lexical, grammatical and punctuation knowledge about a language, which is used to form sentences. Because of their language knowledge they are advanced and less prone to error. For example if a sentence is: "There is a cat sitting on the sofa." But if the number of cat is plural, then these systems are able to change the sentence to: "There are two cats sitting on the sofa."

   KPML [KP] is a NLG Tool based on the Systemic Functional Grammer theory of Halliday [**?**]. It extends the grammer from structural to functional perspective. KPML provides a graphical interface to create templates and grammer rules.

   NaturalOWL [DL] is a NLG Tool based on OWL ontologies. The input to tool is a .owl files which has classes and their information. This information is presented by NaturalOWL in the form of text.

   SimpleNLG [GR09] is a realizer based open source Java API. It is originally developed by Ehud Reiter and research group at the University of Aberdeen. SimpleNLG is a realization engine, which takes structural parts of a sentence as input and combines them to form sentences or paragraphs or sections. It also has lexical and grammatical knowledge of English language, which validates the generated text. More information and tutorials are available on their github page[2]. During analysis of SimpleNLG it is discovered that it: can be easily plugged in any java application and is very easy to learn.

3. **Machine Learning-based**: Machine Learning-based generators are latest in technology, but setting them up requires a huge effort. These systems need to be trained with a huge number of relevant documents and only then these trained NLG system are ready for use. But still after a long time of training with large amount of data, the user is not able to control each and every word of the generated text. Natural Language Toolkit (NLTK) is a python library with NLP and NLG features. However during evaluation of NLTK, it shows that it produces random text

[2]https://github.com/simplenlg/simplenlg

which does not have any sentence structure. It needs to be trained with large data before usage. In addition, the developer does not have power to replace specific words present in generated text.

3. Requirements Analysis

3.1. Requirements Specifications

For the development of tool (pilot project), the standards and methods of *Object Oriented Software* are followed, defined in book ˜Object Oriented Software Engineering Using UML Patterns and Java˝ [BD10]. During the specification phase, several meetings were conducted with experts to understand the requirements and business logic. These meetings helped to understand the requirements of the tool and to formulate concrete functional and non-functional requirements.

3.1.1. Input Attributes

Input to the pilot project (NLG Tool) consists of risk figures, which are collected in excel sheets by *Regression Testing*. The excel sheets contain values of two consecutive runs: Base Run and Regression Run. Rows of excel sheet represent one instrument; this can be an investment or a fund. Columns in the row show different risk values for that instrument. Following are names of few instruments (rows):

- BANK ABC FIX 1.234% 30.04.2019: This represents fund of BANK ABC which has fixed interest rate of 1.234% and it is valid till 30 April 2014.
- EUR_CASH: Cash amount in Euros.
- Y03_ZCB_221: Zero Coupon Bond with three years maturity.
- VaR 99.5% (Sum): Value at Risk for sum of all interest rate with 99.5% confidence level.

These instruments have different categories like: "instruments with fixed interest rate", "instruments with variable interest rate", "instruments with VaR" and so on. Each instrument (row) has many risk figures (as columns) which are calculated by the internal risk model. Following are few columns name:

- VaR 99.5% SUM: VaR (Value at Risk) [Inc] at the confidence level of 99.5% if all individual risks are changed.
- VaR 99.5% IR: VaR at the confidence level of 99.5% if only interest rate relate factors are changed. sum of all individual risks..
- MtM Dirty Value: MtM means Market to Market value [Inb], it symbolizes the way this risk value is calculated. Dirty value means the market prices plus accrued interest until that time.

These columns are also categorized into: "interest-related", "equity-related" and so on.
Experts usually compare data from the current run (Regression Run) with the previous run (Base Run), e.g. risk figures from the current quarter can be compared to predicted risk figures for the next (future) quarter to foresee their impact.

3.1.2. Expected Output

For the pilot project, the scope of data analysis is kept limited, since the main goal is to design and implement an architecture which can support both data analysis and NLG with open source tools. Therefore, the expected output is a textual summary of the analysis. The tool is aimed to check the number of instruments (or entities) which fall into a particular category and columns of that category which have shown deviation (in risk figures) from the previous run. If an instrument shows deviation, it should be counted as affected instrument. In the end, the tool should be able to analyze how many instruments fall into a particular category and how many of them have deviation and the maximum percentage of deviation. A summary of the expected output is shown in Table 3.1.2.

As per expert's requirement, the output text should also depend on the percentage of instruments affected. If the percentage of affected instruments is

Total number of instruments	**Total number of affected instruments**	**Output text**
0	Not applicable	There are no interest rate related instruments in the data set.
1	0	There is one interest rate related instrument in the data set which shows no deviations.
1	1	There is one interest rate related instrument (instrument name) in the data set and it is affected. The maximum deviation observed is x%.
>1	0	There are ___ interest rate related instruments in the data set which all show no deviations.
>1	1	There are ___ interest rate related instruments in the data set. Out of these instruments only «instrument name» is affected. The maximum deviation observed is x%.
>1	>1	There are ___ interest rate related instruments in the data set. Out of these instruments __ instruments are affected which corresponds to y% of all interest related >1 (A few/Almost All/All) instruments. The maximum deviation observed is x%.

Table 3.1.: NLG requirement

between 15 and 85 percent, then experts does not consider it relevant for analysis, hence the output text should be "No Pattern Found.", as shown in Table 3.1.2. The text output also differs for special analysis cases, e.g. in the case of the Monte Carlo Scenario, the user wants to check how many instrument's scenarios have been changed. The expected output is shown in Table 3.1.2.

3.1.3. Functional Requirements

Use case diagram is a type UML diagram, which helps to represent the functional requirements and prospective users of the system to be developed [di].

Percentage of affected instruments	**Output text quantifier**
= 0%	No
> 0% and < 15%	A few
>= 15% and < 85%	No pattern
>= 85% and < 100%	Almost all
= 100%	All

Table 3.2.: NLG quantifiers requirement

Number of instruments with different scenario	**Output text**
= 0	No sentence in output.
= 1	For instrument «instrument name» the used Monte Carlo scenario is not the same.
> 1	There are ___ instruments for which the used Monte Carlo scenario is not the same.

Table 3.3.: NLG requirement for scenario change

It has actors, which are the end-user roles. It has use cases, which represents the functionality of the system. Actors are joined to use cases with an arrow, which represent that actor should have the respective functionality. The functional requirements for the NLG Tool are shown in Figure 3.1. As shown in the use case diagram, the proposed NLG Tool has only one user *Expert* and it should have following functionalities (use cases):

1. **Upload excel data sheet** The NLG Tool should allow the experts (user) to upload an excel sheet, which she wants to analyze.

2. **Select rulesets to run** Data of sheet should be analyzed using business rules. In addition, the expert should be able to select the business rules she wants to run on current data.

3. **Edit data analysis rulesets** The experts should be able to edit the business rules of data analysis.

4. **Edit output text** The expert should be able to edit the output text.

5. **View generated text** The result of the analysis should be expressed in the form of a textual report.

6. **Download generated text file** The generated textual report can be downloaded by the user.

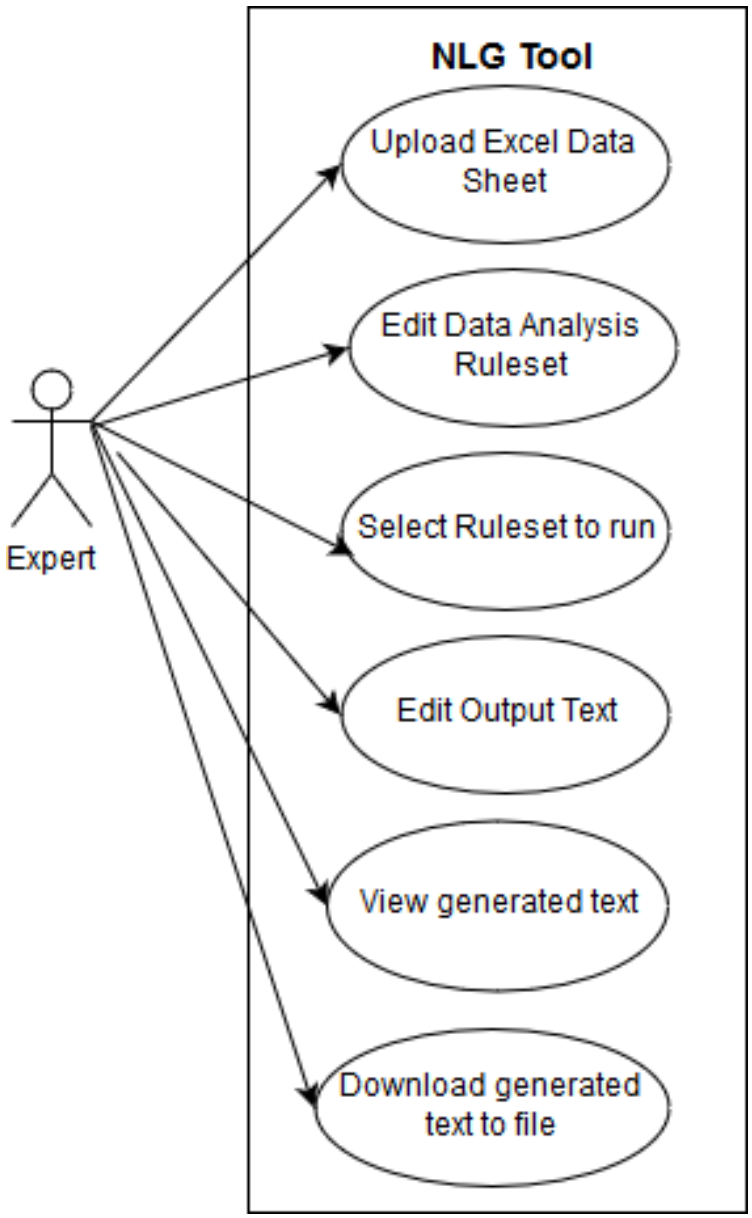


Figure 3.1.: Use Case Diagram

3.1.4. Non-Functional Requirements

Non-Functional Requirements (NFR) of a system are those requirements, which specify the attributes it should have; and constraints it should follow [Gl07]. The attributes of the system include: performance requirements like behavior, speed, timing, etc.; and quality requirements like security, portability, etc. NFRs help to design the architecture of the system to maintain its quality . They are not specific to some behavior of the system, rather help to make the system more user-friendly and robust [Wic]. Apart from the functional requirements, the proposed NLG Tool also has the following NFR:

1. **Simple and Intuitive** The NLG Tool should make the data analysis process simpler. It should reduce the complexity of the process and increase the understanding of data to experts. It should be simple to understand and intuitive to use. Therefore, the business rules should be easy to understand from the perspective of the business user. This will enhance the confidence of experts on decisions taken on the basis of the tool's analysis.

2. **Flexible** The NLG Tool should be flexible enough to integrate different technical plugins. For example, if the business user has the financial data in a MYSQL database, it should take minimal efforts to change the source of data (from Excel Sheet to MYSQL). If the business user wants to use the business rules defined in an other DMN engine, like DROOLS, it should be plugged in with minor efforts. As the tool analyzes a different dataset, with a different number of columns and different instruments, each time the architecture of the tool should be independent of the input data.

3. **User-Adaptable** The NLG Tool should be flexible to allow experts to choose what type of analysis they want to run on a given dataset. It should only carry out the analysis in expert's selected area and the report should contain the text regarding to that analysis. Therefore, the NLG Tool should enable to experts to the following:

    - They can control the tool by selecting which type of analysis should be conducted on the current data sheet.

- They can change the business rules for analysis without the help of developers. This is the most difficult requirement to fulfill, but this research is aimed to give users some degree of freedom.
- They can manipulate some parts of the generated text part without the help of developers.

3.2. Analysis

During the analysis phase, the functional and NFR are analyzed to develop the model of the system [BD10]. According to the gathered requirements, it is clear that the NLG tool will be consist of a Rules Engine and a Natural Language Generator. Therefore, at first, Rules Engine and Natural Language Generator are shortlisted according to the gathered information in research phase.

3.2.1. Rules Engine

In the research phase, two widely used rules engines are discussed: (i) Drools Engine and (ii) Camunda DMN. Both of them are capable of providing the required simplicity and complexity of tool, so one of them is shortlisted based on the following reasons:

- Camunda can be used in a cluster [BD16], therefore it is capable of processing large volumes of data efficiently.
- As the financial experts (which belong to our Regression Testing case) have prior experience with Camunda, so they are familiar with this tool. In comparison they do not know Drools engine, and it takes a significant learning time to get used to a new rules engine.
- In future, they might incorporate this tool with their existing business processes which are built on Camunda BPM. Therefore, it is favorable to use Camunda for development of this NLG tool.

3.2.2. Natural Language Generator

As shown in the research phase, there are many different open-source natural language generators. Out of them *realizer-based natural language generators* are shortlisted for this use case, based on the following reasons:

- Machine Learning-Based: They need a large set of corpora regarding the use case. It is also mentioned by Reiter and Mellish (1993) [RM93], that with the deep NLG systems cost of preparing data and knowledge base is sometimes higher than the benefits gained by them. In this case, training data is not available, therefore, these types of techniques are not useful. Due to the nature of the use case (which is purely business), there should be a control over each word present in a sentence, which is difficult with machine learning-based natural language generator.
- Template-Based: As these types of NLG systems do not have a grammatical check before final text generation, they are very inflexible. Due to their high maintenance nature and lower quality output, this approach is also not favorable for this use case.
- Realizer-Based: This approach does not need huge input data and it allows to have full control over each word present in generated text. Plus, it has grammatical rules to check the generated text. Therefore, this approach is shortlisted for further steps.

After deciding on the type of *natural language generator* which can be used for this tool, the next step is to shortlist one library which is best fitted. Under *Realizer-Based NLG* there are two types of libraries:

- One type, which needs a complete knowledge-base to generate text. For example, NaturalOWL needs a set of classes and objects and their relationship to generate their description. These kind of NLG systems take a lot of time for the intial setup [RM93].
- Other type, which just needs information about the sentence structural components like noun, verb, quantifier, etc.

As the second option directly takes textual information and is simpler to use, therefore they are lighter and more suitable for this use-case. They also addresses two concerns of experts: (i) they do not want to spend a lot of time to

set up the NLG system, (ii) they want to reuse the same tool with few modification to other similar use-cases, therefore this tool should be loosely-bound with data. But this approach also has a drawback, it needs support to plan the content of the document. For the planning of the content we can a use rules engine because [RM93]:

- The user's tasks are limited.
- The user is comfortable with business rules.

In realizer-based approach, *SimpleNLG* is a java-based library and is very easy to use. Therefore, SimpleNLG is shortlisted as NLG-part of tool.

3.2.3. System Model

The next step is to develop an abstract system model. This is done by using two types of UML diagrams: static and dynamic diagrams. *Component Diagram* is used to show the static model and *Activity Diagram* is used to show the dynamic model of tool.

3.2.4. Static System Model

The static system model allows to define the preliminary basic structure of the system. According to the specifications, the tool should be flexible for changing different plugins or APIs for the database and business rules. So, the system is modeled with the following different modules, as shown in Figure 3.2:

- **CamundaProcess**: It acts as the main thread of the tool, which runs the flow of tool and decides on the sequence of tasks.
- **DataInterface**: It is an interface between the main process and the **DataManagement** module. It is responsible for querying data from data source.
- **RulesInterface**: It is an interface between the main process and **CamundaDMN** module. It is responsible for running selected business rules on current data.

- **NLG Interface**: It is an interface between main process and **SimpleNLG** module. Its responsibility is to generate the textual output.

This model allows to incorporate different APIs in different modules of the tool; hence, it decreases dependency and increases flexibility of system.

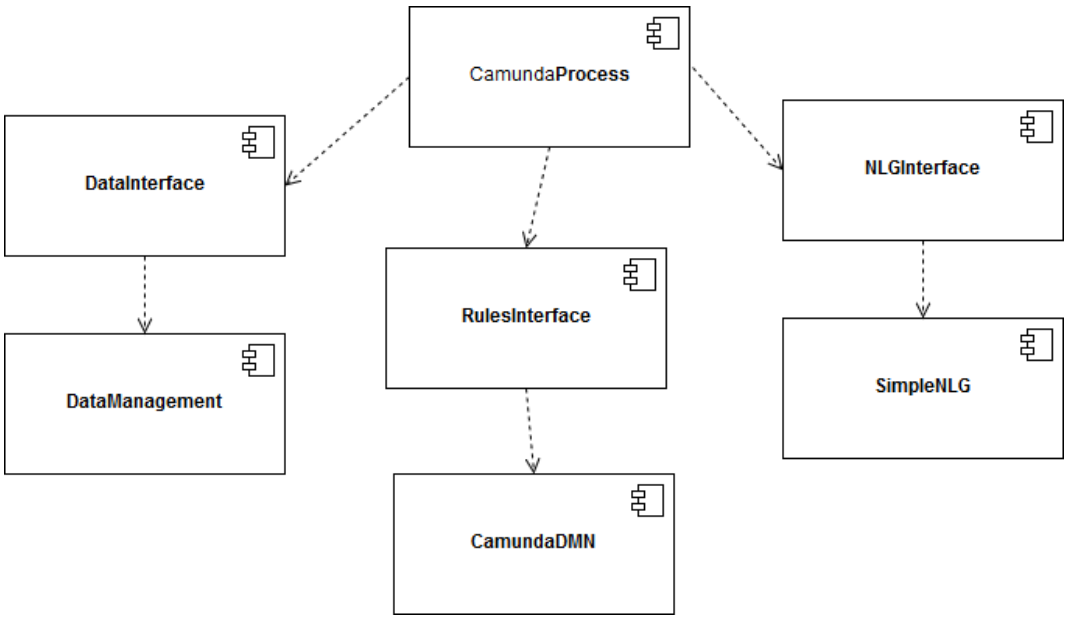


Figure 3.2.: Static Model: Component Diagram

3.2.5. Dynamic System Model

These types of models allow to plan the interaction between different objects. The Activity Diagram as shown in Figure 3.3 explains the expected interaction of experts with the tool. At first, the expert uploads the excel workbook that needs to be analyzed. Then the tool saves the workbook in the server's memory and shows the screen to select the analysis rules that should be run on current data. Further, the user selects the business rules and the tool runs selected rules for the analysis. The result of the analysis is explained in the form of text and shown to the user. Here, the user can see the textual report of the data analysis and save the report. Later, the user can further analyze the data himself and modify or add text to the report.

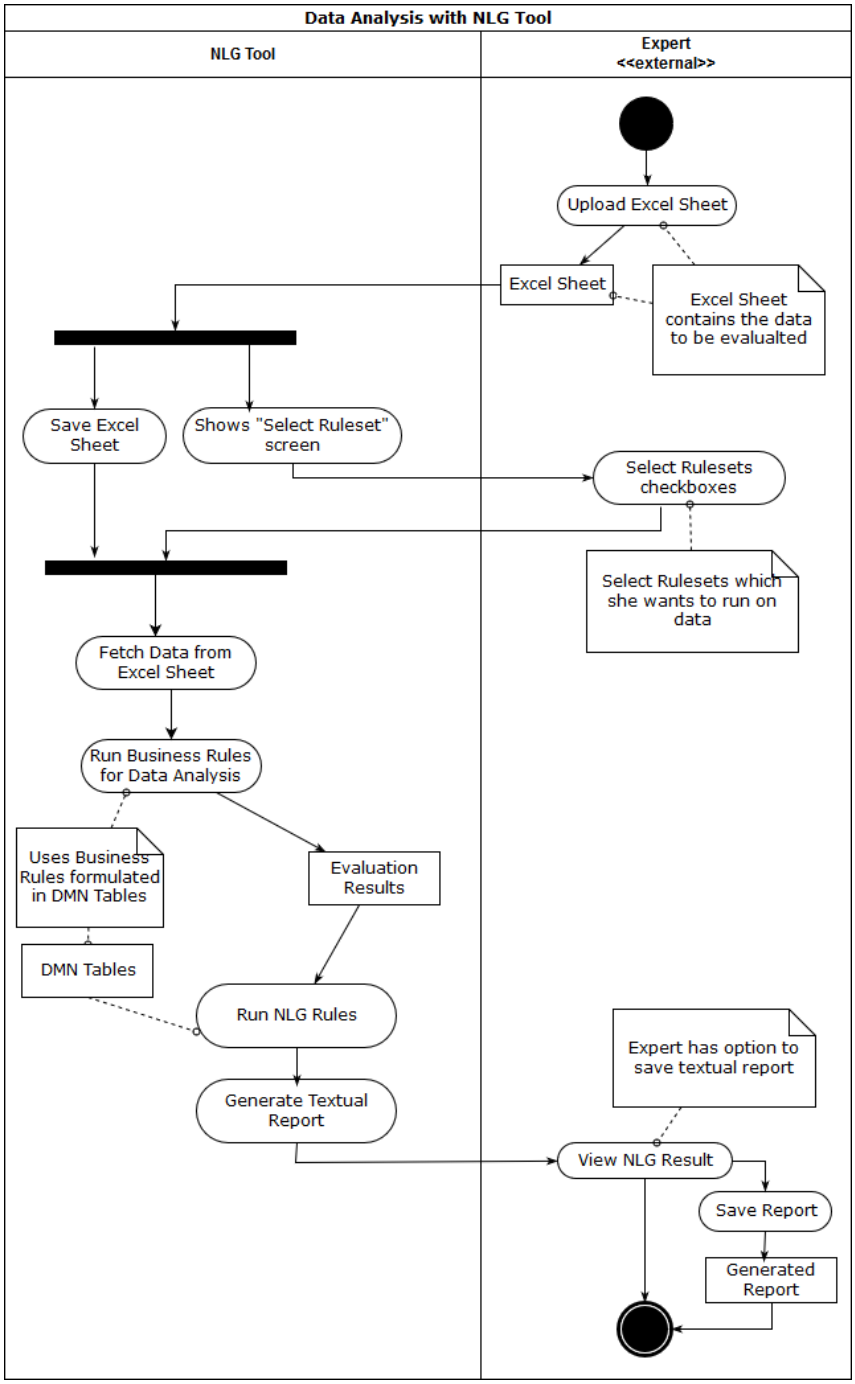


Figure 3.3.: Dynamic Model: Activity Diagram

4. Design and Implementation

4.1. Design

During the design phase, the technical specification of the system starts, as in this phase, detailed technical models of the system are developed. As explained in previous sections, this project objective is to develop a NLG Tool, which allows the user to analyze data and express the analysis result in a textual report. In this section, the implementation of NLG Tool is explained, including the basic project structure and its functionality.

4.1.1. Architecture

Here, the general architecture of *NLG systems* is explained, based on which this NLG tool's architecture is designed. According to Reiter and Dale [ER97], a NLG system consist of three pipelined stages as follows:

1. **Text Planner** As the name suggests, the first stage plans the content of the output text. Which points or information should be covered in the output text?

2. **Sentence Planner** This stage selects the lexicon and plans the sentence structures.

3. **Linguistic Realiser** According to the sentences's plan and grammatical knowledge, the text is generated.

In 2007, Reiter presented an architecture of *data-to-text NLG systems*, which was an extension of the previous NLG system architecture. The data-to-text NLG systems are a type of the NLG systems which process raw data and then generate textual summaries [Re85]. In this NLG Tool architecture, concepts

of the above two architectures are followed. This tool's process has four stages which are described below:

1. **Data Analysis** As the name suggests, in the first stage, the user uploads data to analysis. The uploaded data is analyzed and the results of analysis are forwarded to the next stage. It is made user-adaptable by letting the user select the rules she wants to run on the current data.

2. **Data Interpretation** In this stage, according to the analysis results, the content of the output text is planned. This part is also developed within the Camunda decision table as part of *nlg rules*. It is made user-adaptable by letting the user modify the threshold of the analysis result, which decides the content that should be present in the text output.

3. **Document Planning** This part is also developed within the Camunda decision table and it lets the user select the word choices and logic to select the type of sentence sets she want to develop.

4. **Linguistic Realization** This part is developed with the help of the SimpleNLG library. Data from the Sentence Planning phase is forwarded to SimpleNLG, which generates the desired paragraphs and forms the complete report.

4.1.2. Business Process

The architecture explained above is designed in Camunda BPM, as shown in Figure 4.1. It contains different steps of architecture in form of tasks of a business process. The tasks defined in the NLG Process are explained below:

- *Select Rules to Run*: This task is a "human task", as it asks the experts to select the rulesets which should be used to analyze the current dataset.

- *Fetch Data*: This is a "machine or automatic task". Based on rulesets selected by experts, it queries data from the data source and runs corresponding business rules on them. At the end of this task, the process has results of all rulesets analysis. These first two tasks in combination form "Data Analysis" stage of architecture.

- *Run NLG Rules*: This is also a "machine task". In this step, the analysis results are forwarded to "NLG rules", which then select the sentence structure and the lexicon of the output. It acts as "Data Interpretation and Document Planning" stages of the architecture.
- *Generate Text*: This is a "machine task", which collects the text's information from previous task and generates the "output text". This task is the implementation of "Linguistic Realization" stage.
- *Show generated text*: This is the final step of the process, it is a "human task". Here, a screen with output text is shown to the user with an option to save the generated text.

Figure 4.1.: Business Process

4.2. Implementation

In this sub-section, the technical details of the implementation are explained. The end product of the development is represented in the form of UML diagrams.

4.2.1. Development Environment

The NLG Tool is a Java-based Camunda API, its development environment summary is shown in Table 4.2.1. It is developed using a Camunda BPM, Camunda DMN and SimpleNLG library. It uses the Apache Metamodel libraries for querying excel sheets data. Maven is used to build the deployable version of the application. Further it is deployed on Camunda-Tomcat version 7.0.6. It is tested on Windows 10 platform.

The NLG Tool is developed in incremental iterations. Its development is carried out in the following two stages:

Software/Tool/Framework	Role/Purpose
Apache Maven	Build Management Tool
Apache Metamodel	Library to query excel sheets
Camunda BPM	Business Process Engine
Camunda DMN	Rules Engine
Camunda-Tomcat	Web Server
Eclipse	Integrated Development Environment (IDE)
SimpleNLG	Library for realization of text

Table 4.1.: Development environment of NLG tool

1. **Java Application** : First priority of project was to develop an end to end application which consumes the current data and run business rules, further generates sentences and displays them. In this stage, data from excel sheet is inserted into MongoDB, then this data is fed to a Camunda decision table (based on DMN 1.1). Decision tables contain the basic business rules which analyze the data, then the result of analysis is forwarded to a decision table corresponding to NLG part, which helps in the formation of logic for sentences and gives the lexical choice to user. Then that sentence formation data is passed to the SimpleNLG realizer, which takes the data and generates the desired text. The generated text is displayed on the console output.

2. **Web Application** : This stage provides an user interface to the previous java application. MongoDB is removed as data source and the user is allowed to upload the excel sheet that she wants to analyze. Then she can select the ruleset she wants to run on dataset. Then the decision tables are looped according to the selected ruleset. New decision tables for other types of rulesets are also added, e.g. Scenario Rules. Then it follows the same steps as defined in the previous stage. In the last step, the user is able to see the generated text on the user interface and it also allows the user to save the generated report, so she can continue deeper analysis on data on the same report.

4.2.2. Software Packages

This NLG Tool contains following packages and classes, which are also shown in 4.2:

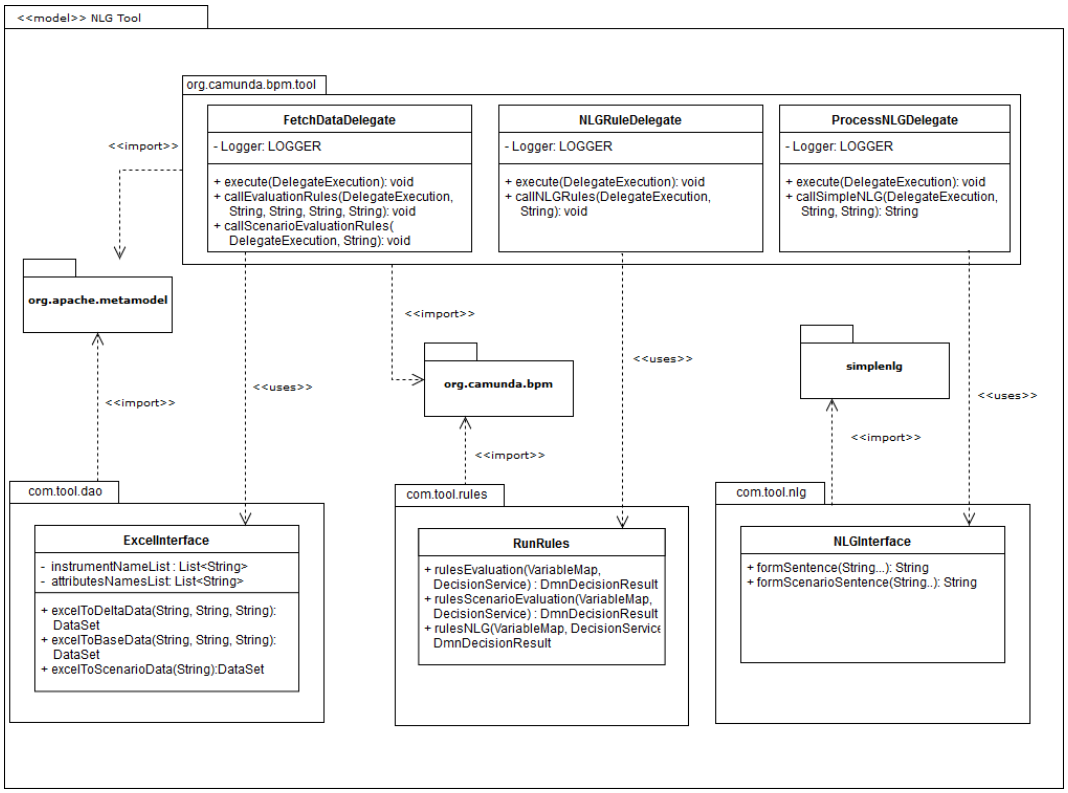


Figure 4.2.: Package Diagram

1. **Package: org.camunda.bpm.tool** This package contains the classes responsible for the execution of the business process. As shown in Figure 4.1, the business process has 5 stages: *Select rules to run, Fetch Data, Run NLG Rules, Generate Text* and *Show generated text.* It has corresponding screen and java classes responsible for the execution of each stage.

    - **FetchDataDelegate** is responsible for the extraction of data corresponding to the current ruleset from the uploaded excel sheet. It also calls the respected decision tables and runs the analysis rules on the extracted data.

- **NLGRuleDelegate** collects the analysis data from the previous stage and runs the rules corresponding to the *NLG*. According to the user defined rules it helps in deciding the sentence structure for the text output. It is a part of *Text Planning* step.

- **ProcessNLGDelegate** collects the data from the previous stage and with the help of *SimpleNLG* library it generates text. It is a part of *Realizer* step.

2. **Package: com.tool.dao** This package handles the communication of API to the datasource. It can have multiple classes each for one type of source. In Java application, it had classes to interact with MongoDB, however in Camunda BPM API it has classes which query data from excel sheets.

   - **ExcelInterface** uses "org.apache.metamodel" api and queries data to excel sheets corresponding to the ruleset to be executed.

3. **Package: com.tool.rules** This package contains the business rules module. It has the following class:

   - **RunRules** uses "org.camunda.bpm.dmn" api to call the business rules. Business rules are configured in camunda decision tables, thus it has information about the decision tables used for the analysis of data.

4. **Package: com.tool.nlg** This package contains the *text generation* module, which has following component:

   - **NLGInterface** uses "SimpleNLG" plugin and generates text.

5. **Imported Packages:** There are few packages which are imported externally and used in various stages of the business process:

- **SimpleNLG**: It is a java library used for *text generation*.
- **org.camunda.bpm**: This java plugin is used to design and execute business rules and business process.
- **org.apache.metamodel**: This java plugin queries data present in the excel workbook. In the first stage of implementation, MongoDB plugin was used to fetch data from MongoDB source.

4.2.3. User Interface Navigation Flow

This sub-section shows the final implemented user screens in the windows web application. Every screen is presented with a screenshot of the running web application with test data. The web application is bundled with Maven with name "nlg-tool-0.1.1.war" and is deployed in "camunda-bpm-tomcat-7.6.0". For reference to the classes and package see 4.2.2.

Screen 1: Home Screen First Camunda-Tomcat is started, then *username: demo* and *password: demo* is used to login. After login, on selection of *Cockpit*, the list of running processes is shown Figure 4.3.

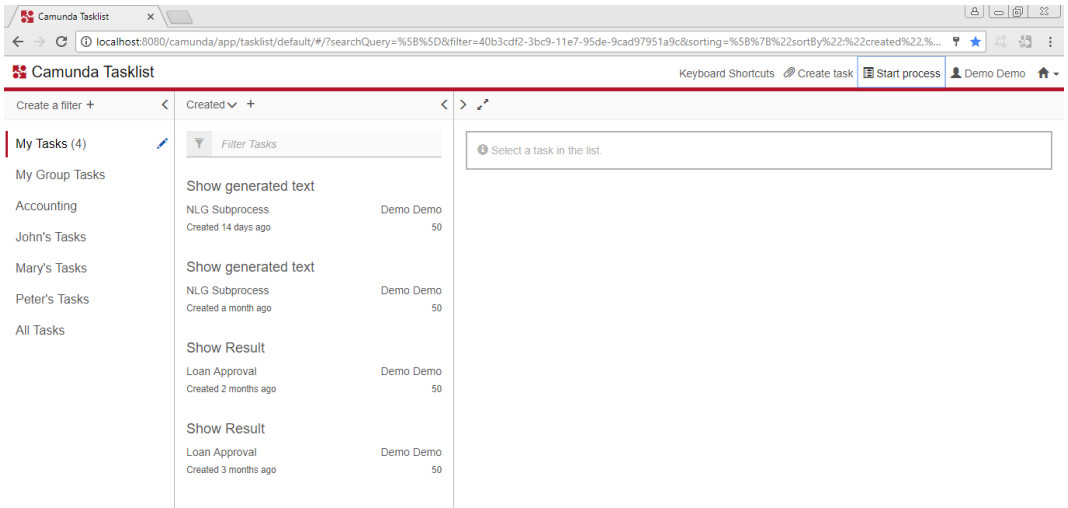


Figure 4.3.: Screen 1: Home Screen

Screen 2: Start Process On clicking on "Start Process" link, screen is displayed as shown in Figure 4.4. "NLG Process" is the Camunda process created for current run.

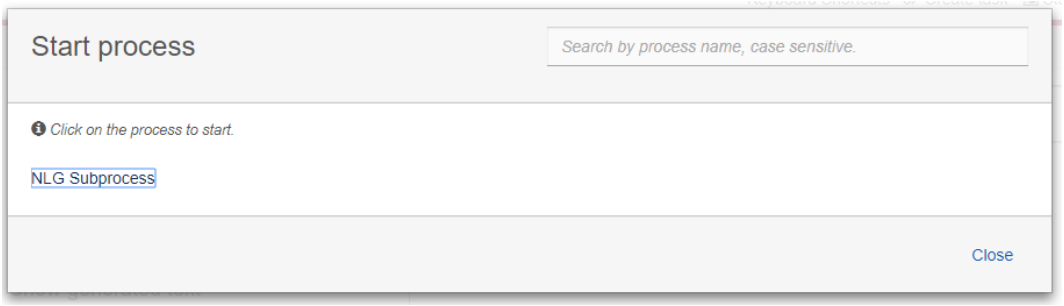


Figure 4.4.: Screen 2: Start Process

Screen 3: Choose File When the process is selected, a new screen is prompted as shown in Figure 4.5. Now the excel data workbook for analysis is uploaded. As for every run the experts want to analyze new data, they are asked to upload first the data they want to analyze. After the excel workbook is uploaded, the user has to click on "Start" to start the process-instance.

Figure 4.5.: Screen 3: Choose File

Screen 4: Select Rules Once the process is started, the next step is to select rules to be run on the current excel file. Screen as shown in Figure 4.6, displays a list of available rulesets which can be used to analyze data.

Screen 5: Show Generated Text After all analysis and text generation, the analysis result is shown in form of a textual report, as shown in Figure 4.7. Here, the user is allowed to download the generated report by clicking on "Download" button.

One sample of downloaded text file is shown in Figure 4.8

Now, the navigation and control flow of the tool is explained with the help of 4.9. As in the development of the tool a modular approach has been followed, every module is divided into separate task of business process, shown

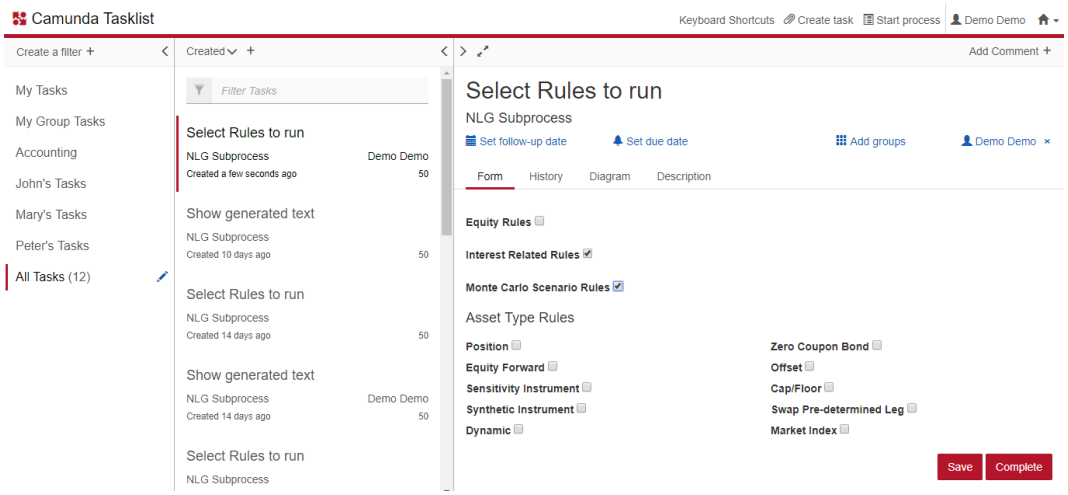


Figure 4.6.: Screen 4: Select Rules

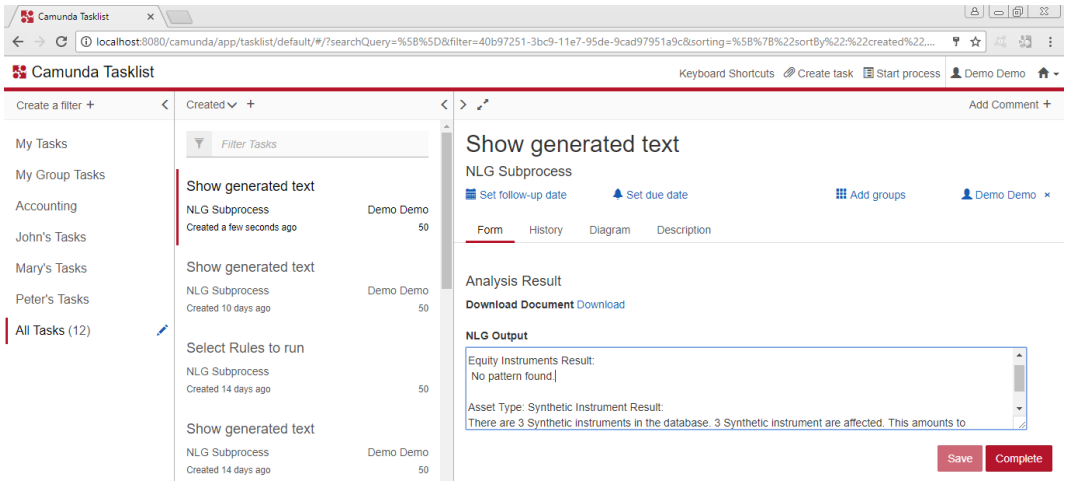


Figure 4.7.: Screen 5: Show Generated Text

in 4.1. The business process acts as the main controller of the tool workflow and each task present in it is responsible for a specific step which is coded in corresponding *Delegate Classes*. At first, when the user uploads the excel file it is forwarded to *FetchDataDelegate*, which collects the file and saves it on the server and waits for the user's response for rulesets selection. Once the user has selected the rules, it queries the data according to the selected rulesets using *DataBaseInterface*. Once the data is ready, this delegate calls *BusinessRulesInterface* and runs analysis business rules. When *FetchDataDelegate* gets the analysis results, it forwards it to *NLGRuleDelegate*. Then *NLGRuleDelegate* calls *BusinessRulesInterface* to run *NLG rules* and forwards the result to *Pro-*

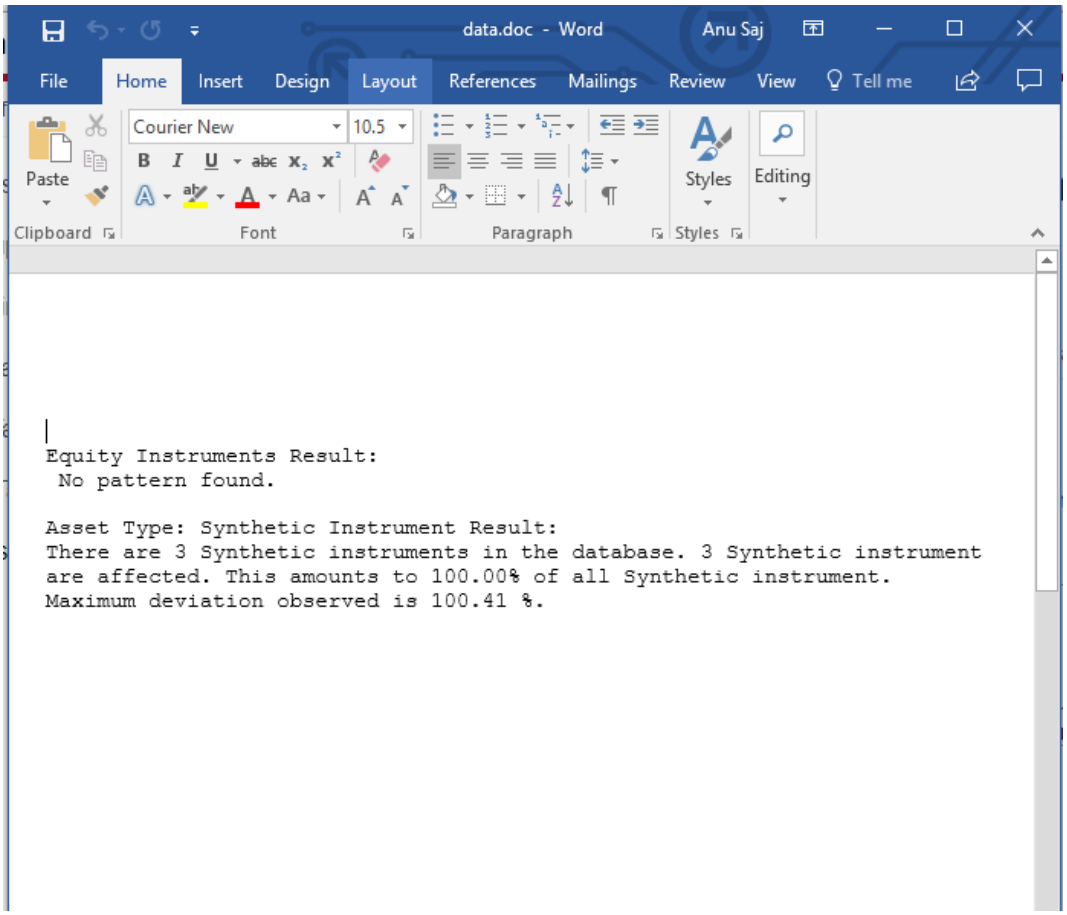


Figure 4.8.: Screen 6: Downloaded Text File

cessNLGDelegate. Then *ProcessNLGDelegate* calls *NLGInterfaceSimpleNLG* plugin to generate text. In the end user previews the generated report.

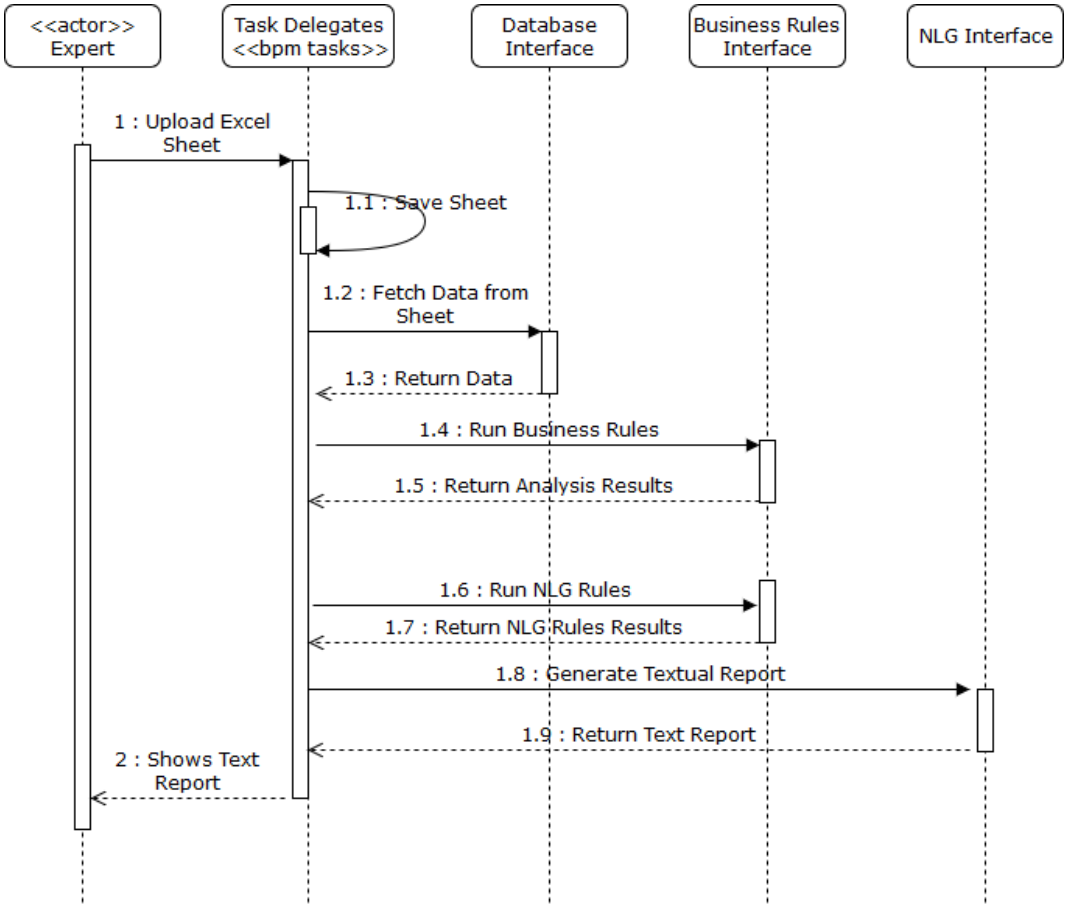


Figure 4.9.: Sequence Diagram

5. Evaluation

5.1. Methodology

The evaluation of the NLG Tool is carried out in two different ways. First, a demo of the NLG tool is given to the financial experts associated with the *Regression Testing of Solvency II* usecase, then they are asked to fill in the questionnaire, as shown in Appendix A. As the number of participants for this questionnaire was limited, so one online questionnaire is also conducted, as shown in Appendix B. The online survey has the following steps:

Step 1: Basic Information : In this section, the participants are asked about their age, gender and experience as a financial expert.

Step 2: Generated Text Evaluation : In this section, participants are asked to imagine that they are financial experts and their task is to analyze the financial figures of different companies. They are provided with an excel workbook, which contains four sheets:

- 2016: Companies' data regarding 2016 year.
- 2017: Companies' data regarding 2017 year.
- Companies_Properties: Every company which belongs to one of different categories like, "Textile", "Construction" and "Machinery". Third sheet "Company Category" has information about the category of company.
- Column_Properties: The fourth sheet has information about the columns. It shows which columns are related to *profit* and which are related to *loss*.

In the first two sheets, every company represents a row and the columns contain the financial figures respective to that company. Now, the participants have to look at the excel workbook and try to find out which category of companies have maximum profit compared to last year. Later, the participants were asked to imagine that the data of sheet increases to 10,000 companies and 300 columns, then they were shown the following text:

Textile Related Result:
There are 500 Textile Related companies in the database. 104 Textile Related companies are profited. This amounts to 20.8 % of all Textile Related companies. Maximum profit observed is 23.12 %.

Construction Related Result:
There are 509 Construction Related companies in the database. 50 Construction Related companies are profited. This amounts to 10 % of all Construction Related companies. Maximum profit observed is 30.02 %.

Machinery Related Result:
There is only one Machinery Related company in the database which shows no profit.
This section has the following statements about the text:

- Above text is easy to read and understandable.
- Above text is helpful in data analysis.
- I would like to see different variation of text generated from tool.
- I would like to see same kind of sentences, as it is easy to find information in similar sentence structures.

The participants responses are collected on seven-level Likert scale, where 1 (left-most choice) represents "completely disagree"; and 7 (right-most choice) represents "completely agree".

Step 3: NLG Tool Evaluation : Here, the participants have to give feedback about the NLG Tool approach introduced in the previous section. Following are the questions of this section:

- Above mentioned tool is helpful.
- This tool will help me to plan deep analysis of data.
- I would like to use this tool for further analysis.
- I like most about the tool.
- I like least about the tool.
- I wish this tool had following feature or functionality.

The first three questions are statements, which are rated on Likert scale by the participants. The other questions are free-text questions, where the participants can describe their experience and recommendations about the tool. The responses of all questions are discussed in detail in Section 5.2.

5.2. Evaluation Results

In this section, the results of both questionnaires are discussed. There have been in total 18 participants, which include 2 financial experts and other professionals from different industries and few students. 33% of participants are female and the rest are male, shown in Figure 5.1. Age distribution of the questionnaire is shown in Figure 5.1. 50% of participants are between 25 to 30 years old, second highest (22 %) age group is between 30 to 35 years.

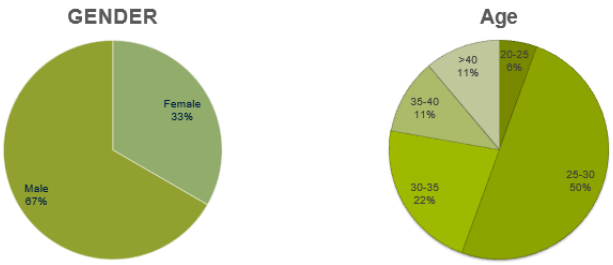


Figure 5.1.: Survey: Age and Gender

Section II of questionnaire has 4 questions regarding the generated text, shown in Section 5.1. The result of each Likert scale-question is represented by horizontal bar chart, as shown in Figure 5.2. In the chart width of "dark green" color shows the percentage of "strongly agree" (7) responses; width of "dark

red" color shows the percentage of "strongly disagree" (1) responses; width of "grey" color shows the percentage of "neutral" (4) responses. The green and red shades are respectively lighter for the responses which are positioned near to "neutral" (4) on Likert-scale.

$\frac{1}{3}$ of participants strongly agree that the generated text is easy to understand. Rest of the participants also agree to some degree that text was understandable, except $\frac{1}{10}$ participants, which find it little difficult, but no reason was mentioned. They also agree that this text will help them with the deeper analysis of the provided data, only 11% are neutral about it. The comment added with the neutral response was the lack of multidimensional analysis by the tool. Regarding the variation of generated text, the responses were mixed, as some of them would prefer different formulation of sentences. Regarding the preference of one type of text (without variation) with every report, many participants agree that it is easier to find information from same sentences structures.

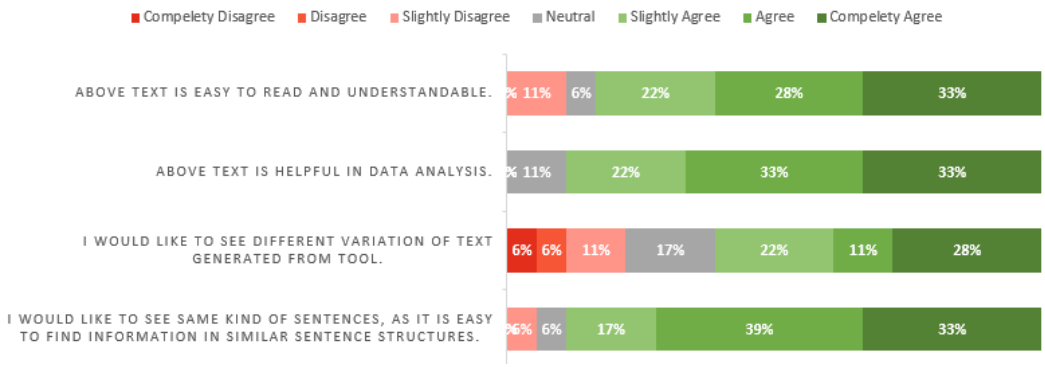


Figure 5.2.: Survey: Questions about Text

Section III of the questionnaire is focused on the NLG Approach employed in this thesis, the questions are shown in Section 5.1. Their results are presented using the same kind of horizontal bar chart, shown in Figure 5.3. When participants were asked whether the tool is helpful, the majority agree with it except one participant who gave neutral response. Similarly, the majority of them believe that this tool will help them to focus on deeper analysis of the data, as they will get a quick summary of the data beforehand. This tool will help them in saving time. Except one participant with negative response and 2 participants with neutral responses, were not sure that this tool might help with data analysis. In the end, participants gave a positive response regarding

using the NLG tool for their further data analysis tasks, except 2 people who did not specify any reason.

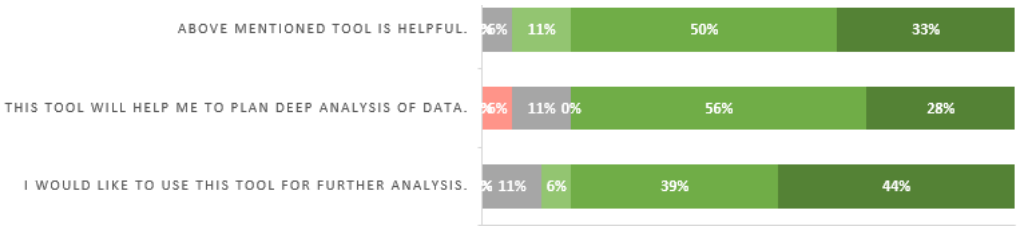


Figure 5.3.: Survey: Questions about NLG Tool

If any conclusion may be drawn from the evaluations, then they are as follows. The NLG approach with data analysis is liked by a significant number of participants. The majority of the participants agrees that this NLG tool will help the financial experts with their data analysis work. Nevertheless, this NLG Tool approach definitely got positive responses, still few of them will take time to accept such a solution.

However, the participants have some suggestions and concerns like: they want to know how this system is configured, and want to see the logic behind the processing of data. Some of them suggest to display data not only in text form, but also using some smaller data tables. Some are worried about the effort they need to put to customize the tool according to their needs.

6. Conclusion

In this thesis, an attempt has been made to tailor a user-adaptable NLG solution for the defined use case using open-source libraries. A NLG Tool is implemented using Camunda BPMN, Camunda DMN and SimpleNLG libraries. It first analyzes data uploaded by the user and then presents the analysis result in a textual report format. Goal of the thesis is to make this tool intuitive and user-adaptable. For achieving this, the following concepts were employed [RM93]:

- As experts are familiar with BRMS, business rules are used to gather requirements.
- The users (experts) can control the tool with *user-interaction*.

User is allowed to do following activities without the interference of programmer:

- Upload data to be analyzed.
- Select the set of rules to be run on current dataset. Only rules which are selected by users run on the data, therefore controls analysis.
- Change rules of analysis (logic to evaluate data).
- Change rules of NLG (logic which selects the sentence structures based on analysis results).
- Change lexicon in output text.

Some limitations of NLG Tool are:

- As SimpleNLG library (realizer) is used as NL generator, only English and German language libraries of it are open source. If user wants to generate text in different language, it is not possible with open-source libraries of SimpleNLG.

- As business rules are hand-tailored, it gives the user the power to change them, but it also needs a significant investment of time and effort from her.
- Although the developed tool is user-adaptable, she cannot use it for a different use case without the help of developer.

In evaluation, the participants gave significantly positive feedback regarding this approach. Although some concerns regarding efforts and cost exists, but the overall response is encouraging.

7. Future Work

In this section, the prospective future work are listed. As this thesis developed a pilot project, it has a strong potential of developing an enterprise-scale project. Few points are taken from financial experts feedback on NLG tool are following:

1. This tool can be extended more for complex analysis of data. It can be done by adding more business rules, this will help to exploit maximum benefit of computing power of machines.

2. Text planning and sentence planning stages could be designed as more user-adaptable by adding more business rules for NLG part. So that user has a wide range options for changing lexicon without help of the programmer.

3. As from feedback from one of our financial expert, more variations of sentences can be added, so that every report generated does not have same set of sentences. It will add variety to the reports and will reduce the experts load of writing reports in different templates.

4. One more feature could be allowing user to define different abstraction level (i.e. detailed level) of report. For example, for higher management report should be concise and only contain few important points. For peer financial experts report should be more detailed covering each aspect of data analysis. This is inspired from "BabyTalk" [Po08].

The possibilities of improvements in NLG systems are mostly dependent on prospective users and the use case under investigation. Therefore, length of this list of future improvements can vary too. For example, one user might wants multilingual text but other does not wants it.
As the capacity of data storage is increasing and prices of memory storage are falling, many industries tend to adapt the data-driven approaches. As data provides a clear picture of past values and helps to take decisions for the future,

therefore soon the demand of DDD technology will increase [Ra]. A survey [BS] findings state that:
"Only one-third of enterprises currently use information to identify new business opportunities and predict future trends and behavior, but most of the remaining two-thirds plan to do so in the future."
According to the survey, the main reasons behind not using DDD are:

- For DDD either data is not available, or quality of available data is poor.
- The organization's culture does not support DDD.

Now, companies are focusing to collect quality data [BS] and researchers are focusing to develop strategies which can help in building the data-driven culture in organizations [Ni17]. With the increasing demand of DDD technologies, need of automated textual report generation tool will also increase. Therefore, DDD demand and the NLG tool requirement go in hand in hand. The need of the NLG tool is expected to increase in the future and it will effect every industry which follows or will follow DDD like: marketing, finance, health, academics, internet of things (IOTs) [Ni], ecology [KC15].

A. Evaluation Questionnaire for Experts

NLG Tool Evaluation Questionnaire

Section I: Basic Information of User

Gender

Please choose only one of the following:

- ○ Male
- ○ Female

Age

Please write your age here:

For how many years are you working as a Financial Expert?

Please write your answer here:

If you do not have any experience as Financial Expert then enter "0" (i.e. zero).

Section II: Generated Text Evaluation

Imagine you want to analyze Risk Figures generated from Solvency II Regression Testing, which are present in excel workbook.

Please use the NLG Tool provided and use it for preliminary data analysis. As a result, NLG Tool generates following output:

"*Equity Instruments Result:*

There are 5 Equity Related instruments in the database. 1 Equity Related instrument is found. This amounts to 20.00% of all Equity Related instruments. Maximum deviation observed is 10.02 %.

Affected instrument is: Position View 1. "

Please answer the following questions regarding text output of tool:

	Completely Disagree			Neutral			Completely Agree
Above text is easy to read and understandable.	○	○	○	○	○	○	○
Above text is helpful for analysis of data.	○	○	○	○	○	○	○
I would like to see different variation of text generated from tool.	○	○	○	○	○	○	○
I would like to see same kind of sentences, as it is easy to find information in similar sentence structures	○	○	○	○	○	○	○

Section III: NLG Tool Evaluation

Please answer the following questions regarding this tool:

(Select the appropriate response.)

	Completely Disagree			Neutral			Completely Agree
Above mentioned tool is helpful.	○	○	○	○	○	○	○
This tool will help me to plan deep analysis of data.	○	○	○	○	○	○	○
I would like to use this tool for further analysis	○	○	○	○	○	○	○

I like most about the tool:

(Please write your answer below.)

I like least about the tool:

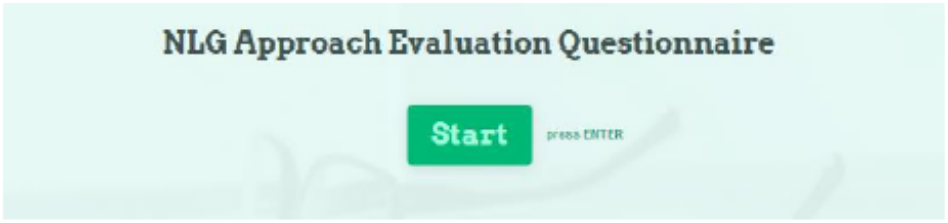
(Please write your answer below.)

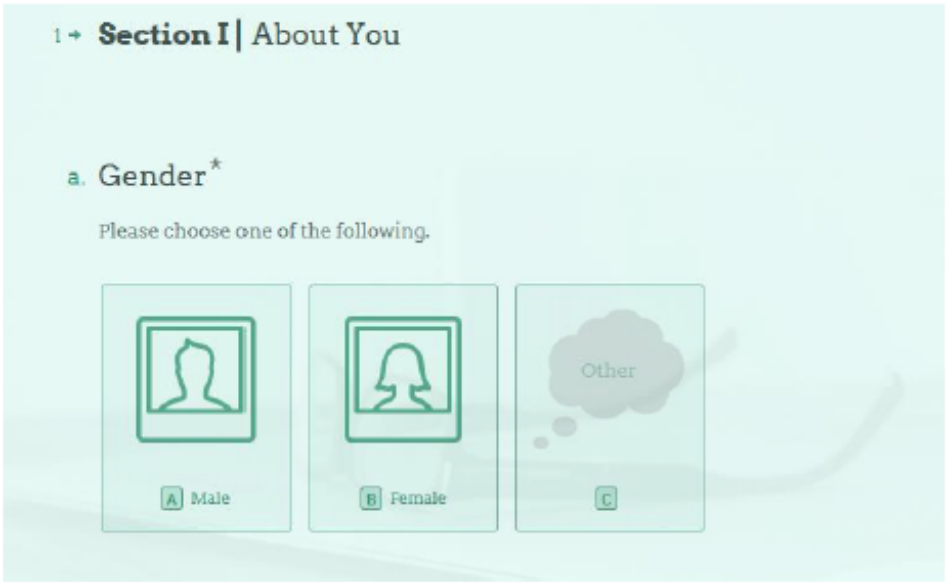
I wish this tool had following feature or functionality:

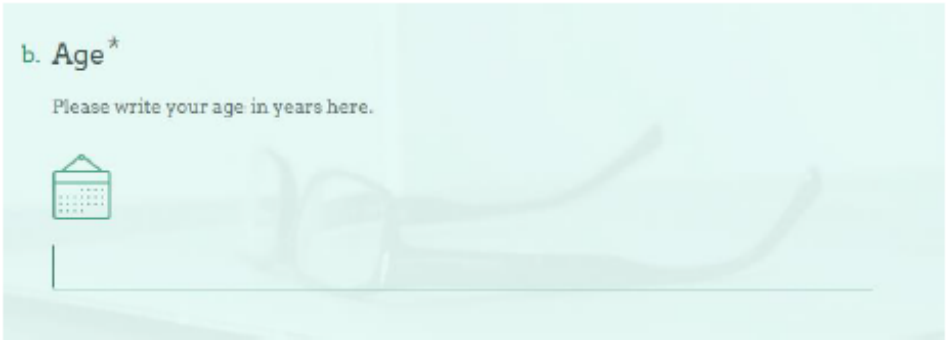
(Please write your answer below.)

B. Evaluation Questionnaire (Online) for General People

Online link for questionnaire: https://anu151.typeform.com/to/ddv3AZ







c. For how many years are you working as a Financial Expert?*

If you do not have any experience as Financial Expert then enter "0" (i.e. zero).

Section II | Generated text Evaluation

Assume you are a Financial Expert and want to check the financial performance of different companies. You are provided with a sheet of their financial figures, which are collected from different resources. To analyze performances of companies you always compare their current figures (i.e. 2017 data) with past figures (i.e. 2016 data).

(To continue further please open the excel sheet provided in attachment.)

The excel workbook has 2 sheets, one for current figures and other for past figures. Both sheet have rows as **companies' name** and columns contains the **financial figures** associated with companies. Each company has different categories, similarly columns have many properties.

Assume your task is to check in which category of companies performed best and what was their maximum profit margin.

(For this task please open the attached excel sheet and try to analyze it for 1-2 minutes.)

Assume that number of rows has increased i.e. number of companies to analyze are 10,000 and number of columns are increased to 300.

Assume there is a software which allow you to upload the excel file and it calculates the difference of old and new values. In addition, its allow you to choose the categories of companies and columns that should be analyzed. As a result, its generates following output:

"Textile Related Result:
There are 500 Textile Related companies in the database. 104 Textile Related companies are profited. This amounts to 20.8 % of all Textile Related companies. Maximum profit observed is 100.02 %.

Construction Related Result:
There are 509 Construction Related companies in the database. 324 Construction Related companies are profited. This amounts to 63.65% of all Construction Related companies. Maximum profit observed is 100.02 %.

Machinery Related Result:
There is only one Machinery Related company in the database which shows no profit. "

Continue press ENTER

2 → **Section II |** Generated Text Evaluation

a. Above text is easy to read and understandable.*

0	1	2	3	4	5	6

Completely Disagree Neutral Completely Agree

b. Above text is helpful in data analysis.*

0	1	2	3	4	5	6

Completely Disagree Neutral Completely Agree

c. I would like to see different variation of text generated from tool.*

0	1	2	3	4	5	6

Completely Disagree — Neutral — Completely Agree

d. I would like to see same kind of sentences, as it is easy to find information in similar sentence structures.*

0	1	2	3	4	5	6

Completely Disagree — Neutral — Completely Agree

3 → **Section III**: NLG Tool Evaluation

Please answer the following questions regarding this tool:

(Select the appropriate response.)

a. Above mentioned tool is helpful.*

0	1	2	3	4	5	6

Completely Disagree — Neutral — Completely Agree

b. This tool will help me to plan deep analysis of data.*

0	1	2	3	4	5	6

Completely Disagree — Neutral — Completely Agree

c. I would like to use this tool for further analysis. *

0	1	2	3	4	5	6

Completely Disagree — Neutral — Completely Agree

d. I like most about the tool:

(Please write your answer below.)

SHIFT + **ENTER** to make a line break

e. I like least about the tool:

(Please write your answer below.)

SHIFT + **ENTER** to make a line break

f. I wish this tool had following feature or functionality:

(Please write your answer below.)

SHIFT + **ENTER** to make a line break

OK ✓ press ENTER

Submit press ENTER

Never submit passwords! - Report abuse

C. AX-Semantics Screenshots

Step 1: Create Project

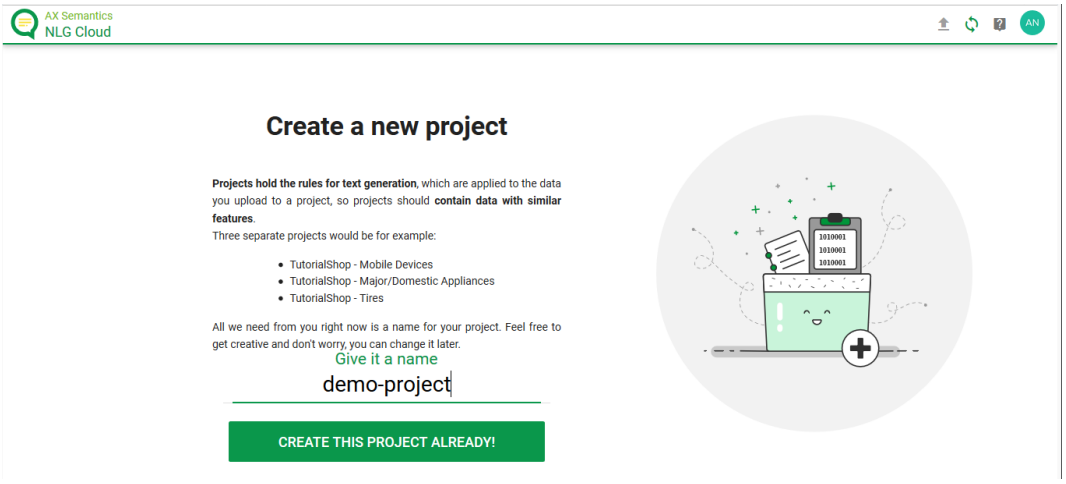


Figure C.1.: Create Project

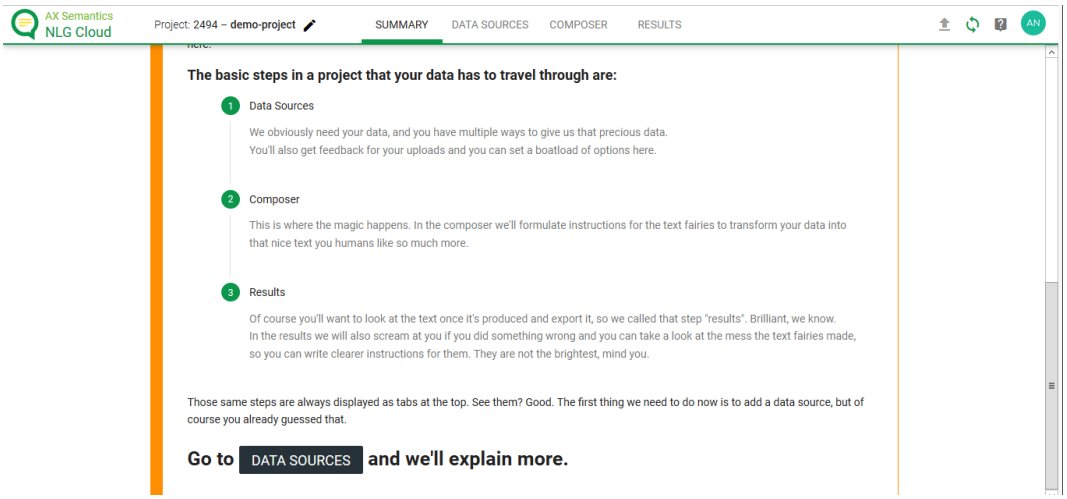


Figure C.2.: Basic Steps

Step 2: Create Datasource

Step 3: Create Text

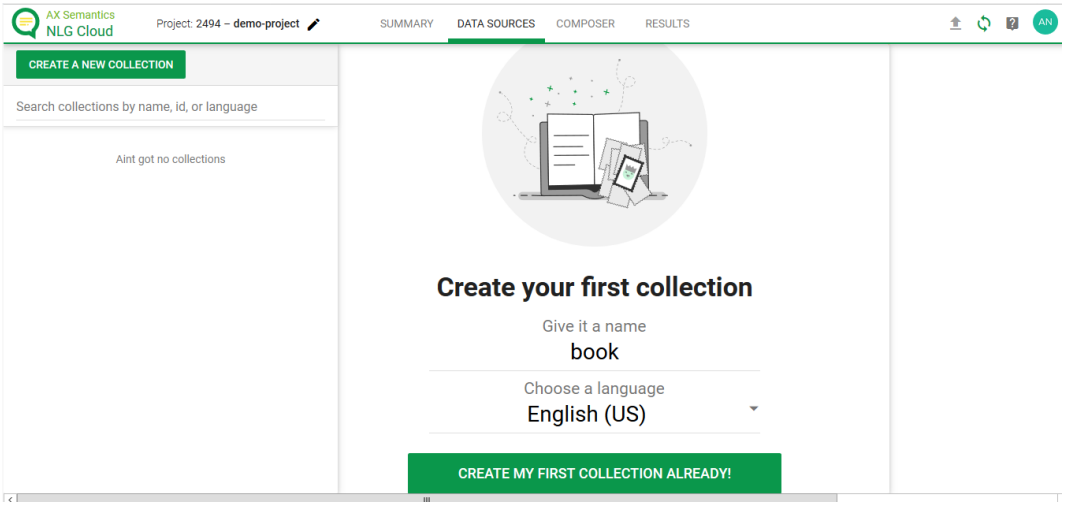


Figure C.3.: Create Collection

	A	B	C	D	E	F	G	H	I
1	**name**	**author**	**year**	**price**	**copies**	**genre**	**goodreads**	**press**	**language**
2	Don Quixote	Miguel de	1605		500 million	fiction (novel)			Spanish
3	A Tale of Two Cities	Charles Dickens	1859		200 million	fiction historical	Y		English
4	The Lord of the Rings	J. R. R.	1954		150 million	fiction: fantasy	Y		English
5	Le Petit Prince (The	Antoine de Saint-			140 million	fiction			French
6	Harry Potter and the Philosopher's Stone	J. K. Rowling	1997		120 million	fantasy	Y		English
7	The Hobbit	J. R. R. Tolkien	1937		100 million	fantasy			English
8	And Then There	Agatha Christie	1939		100 million	mystery	Y		English
9	Dream of the Red	Cao Xueqin	1754		100 million	family saga			Chinese
10	Alice in Wonderland	Lewis Carroll	1865		100 million	fantasy			English

Figure C.4.: CSV File with Data

Step 4: Generate Result

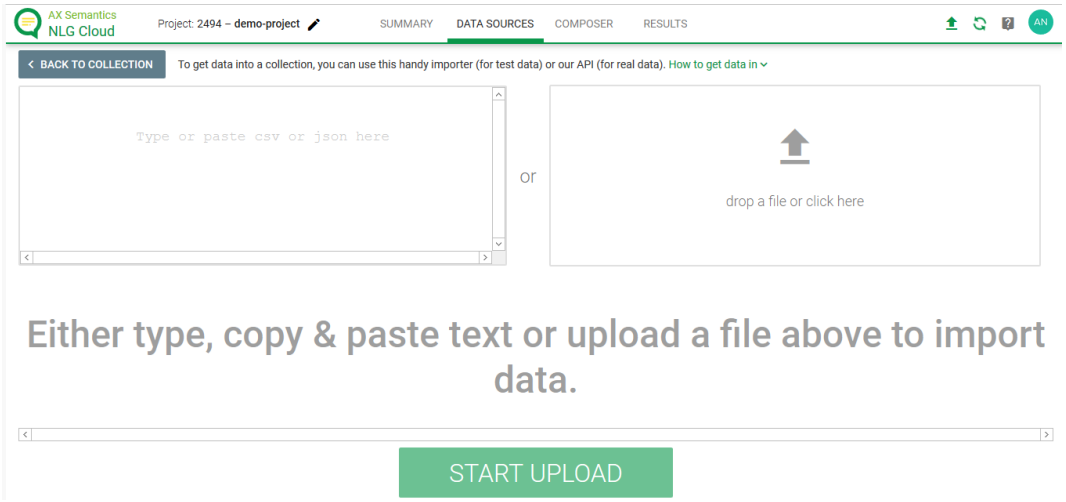


Figure C.5.: Upload Collection Data

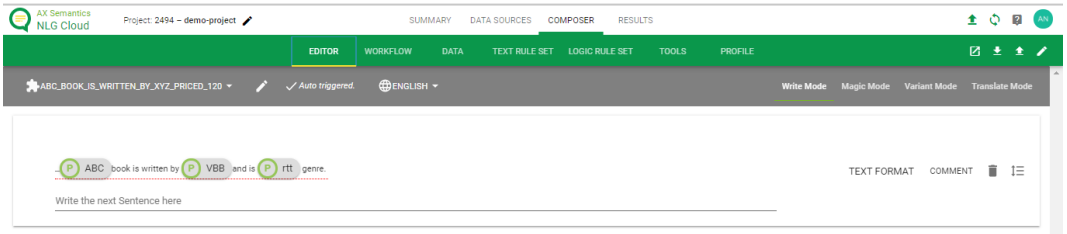


Figure C.6.: Create Text Containers

Figure C.7.: Generate Phrase Containers

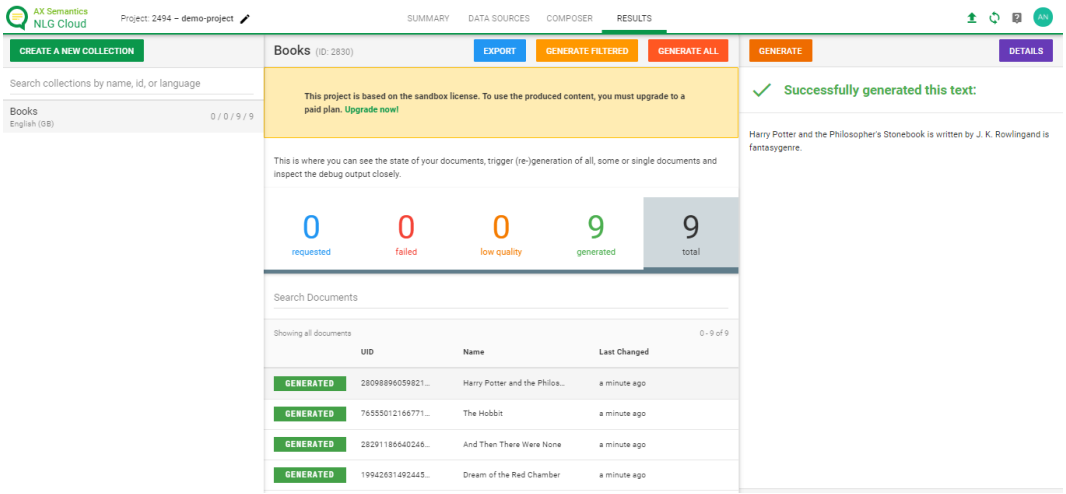


Figure C.8.: Results

Bibliography

[BD10] Bruegge, B.; Dutoit, A. H.: *Object Oriented Software Engineering Using UML Patterns and Java*. 3rd edition. 2010. 3.1, 3.2

[BD16] Bischoff, B.; van Dinther, C.: *Workflow Management Systems-an analysis of current open source products*. In *Lecture Notes in Informatics (LNI), Gesellschaft fur Informatik, Bonn*. Digital Enterprise Computing. 2016. 2.3, 2.3.2, 3.2.1

[Be02] Becker, T.: *Practical, Template - Based Natural Language Generation with TAG. Proceedings of the Sixth International Workshop on Tree Adjoining Grammar and Related Frameworks*. 2002. 1

[Br] Bruner, J.: *Forbes: Five Steps For Making Data-Driven Decisions*. https://www.forbes.com/sites/jonbruner/2012/04/20/five-steps-for-making-data-driven-decisions/#70349b6348de. Last accessed on 07.10.2017. 1.1

[BS] BI-Survey.com: *14 Survey-Based Recommendations on How to Improve Data-Driven Decision-Making*. https://bi-survey.com/data-driven-decision-making-business. Last accessed on 30.12.2017. 1.1, 7

[Bu] Buluswar, M.: *McKinsey: How companies are using big data and analytics*. https://www.mckinsey.com/business-functions/mckinsey-analytics/our-insights/how-companies-are-using-big-data-and-analytics. Last accessed on 14.01.2018. 1.1

[Caa] Camunda: *Camunda BPM Editions*. https://camunda.com/bpm/enterprise/. Last accessed on 01.08.2017. 2.3.2

[Cab] Camunda: *Supported Environments*. https://docs.camunda.org/manual/7.7/introduction/supported-environments/. Last accessed on 01.08.2017. 2.3.2

[Co97] Cole, R.; Mariani, J.; Uszkoreit, H.; Batista, G.; Varile; Annie Zaenen, Antonio Zampolli, V. Z.: *Survey of the State of the Art in Human Language Technology*. chapter 4. Language Generation. Cambridge University Press and Giardini. 1997. 2.1.3

[di] uml diagrams.org: *UML Use Case Diagrams*. https://www.uml-diagrams.org/use-case-diagrams.html. Last accessed on 30.10.2017. 3.1.3

[DKT03] van Deemter, K.; Krahmer, E.; Theune, M.: *Real vs. template-based natural language generation: a false opposition? Association for Computational Linguistics*. 2003. 1

[DL] DL, A.: *Generating natural language descriptions from OWL ontologies the natural OWL system. COMMUNICATIONS OF THE ACM*. 2

[EE] Eckerson, H. H.; Eckerson, W. W.: *KDnuggets: Natural Language Generation overview - is NLG is worth a thousand pictures ?* https://www.kdnuggets.com/2017/05/nlg-natural-language-generation-overview.html. Last accessed on 27.09.2017. 2.2, 2.2.1

[EG94] Eli Goldberg, Norbert Driedger, R. I. K.: *Using Natural-Language Processing to Produce Weather Forecasts. IEEE*. April 1994. 2.1.1

[Ei15] Eid, T. K.: *Solvency II. University of Oslo*. October 2015. 1.2.2

[ER97] Ehud Reiter, R. D.: *Building Applied Natural Language Generation Systems*. May 1997. 4.1.1

[Gi08] Gillespie, O.; Clark, D.; Verheugen, H.; Wells, G.: *Benefits and challenges of using an internal model for Solvency II. Milliman White Paper*. November 2008. 1.2.2

[Gl07] Glinz, M.: *On Non-Functional Requirements. 15th IEEE International Requirements Engineering Conference (RE 2007)*. 2007. 3.1.4

[Gr] Group, O. M.: *Business Process Model and Notation*. http://www.bpmn.org. Last accessed on 30.08.2017. 2.3

[GR09] Gatt, A.; Reiter, E.: *SimpleNLG: A realisation engine for practical applications*. In *Proceedings of the 12th European Workshop on Natural Language Generation*. page 90â93. Association for Computational Linguistics. March 2009. 2

[Ina] Investopedia: *Insurance*. https://www.investopedia.com/terms/i/insurance.asp. Last accessed on 01.09.2017. 1.2.1

[Inb] Investopedia: *Mark To Market - MTM*. https://www.investopedia.com/terms/m/marktomarket.asp. Last accessed on 31.12.2017. 3.1.1

[Inc] Investopedia: *Value At Risk - VaR*. https://www.investopedia.com/terms/v/var.asp. Last accessed on 01.09.2017. 3.1.1

[KC15] Karduck, A. P.; Chitlur, S. S.: *Data driven decision making for sustainable smart environments*. 2015. 7

[KP] KPML: *What is KPML?* http://www.fb10.uni-bremen.de/anglistik/langpro/kpml/kpml-description.htm. Last accessed on 23.05.2017. 2

[LF] Litman, S.; Fernandez, J.: *Measuring Success and GuideStar Exchange: 7 Steps for Data-Driven Decision Making*. https://www.slideshare.net/guidestar/guidestar-september-2010-webinar-9-2910-5363649. Last accessed on 01.12.2017. 1.1

[LR16] Lemon, D. G. O.; Rieser, V.: *Natural Language Generation enhances human decision - making with uncertain information*. page 264 to 268. Association for Computational Linguistics. 2016. 2.1

[MS] Method; Style: *What is DMN ?* https://methodandstyle.com/what-is-dmn/. Last accessed on 03.05.2017. 2.3

[Ni] Nickel, C.; Farr, L.; Wood, A.; Bergman, S.; Cunningham, C. J. L.: *Making it stick: The secret to developing a data-driven culture*. https://www.jisc.ac.uk/reports/

the-future-of-data-driven-decision-making. Last accessed on 30.12.2017. 7

[Ni17] Nickel, C.; Farr, L.; Wood, A.; Bergman, S.; Cunningham, C. J. L.: *Making it stick: The secret to developing a data-driven culture*. University of Tennessee at Chattanooga. 2017. 7

[Or] Org, D.: *Overview*. https://docs.jboss.org. Last accessed on 01.10.2017. 2.3.1

[OR01] Overmyer, Scott ;Lavoie, B.; Rambow, O.: *Conceptual Modeling through Linguistic Analysis Using LIDA*. CoGenTex, Inc. 2001. 2.1.3

[PF13] Provost, F.; Fawcett, T.: *Data Science and its Relationship to Big Data and Data-Driven Decision Making*. *Mary Ann Liebert Inc.* March 2013. 1.1

[Po08] Portet, F.; Reiter, E.; Gatta, A.; Huntera, J.; Sripadaa, S.; Freer, Y.; Sykes, C.: *Automatic generation of textual summaries from neonatal intensive care data*. 2008. 2.1.3, 4

[Ra] Rajeck, J.: *Five trends which will define data-driven marketing in 2017*. https://econsultancy.com/blog/68661-five-trends-which-will-define-data-driven-marketing-in-2017. Last accessed on 30.12.2017. 7

[Re85] Reiter, E.: *An Architecture for Data-to-Text Systems*. 1985. 4.1.1

[RM93] Reiter, E.; Mellish, C.: *Optimizing Cost and Benefits of Natural Language Generation*. *University of Edinburgh*. 1993. 2.1.3, 3.2.2, 6

[Roa] Ross, S.: *Investopedia: What is the main business model for insurance companies?* https://www.investopedia.com/ask/answers/052015/what-main-business-model-insurance-companies.asp. Last accessed on 01.09.2017. 1.2.1

[Rob] Roth, E.: *SISENSE: 5 Steps to Data-Driven Business Decisions*. https://www.sisense.com/blog/5-steps-to-data-driven-business-decisions/. Last accessed on 31.12.2017. 1.1

[Ta] Taylor, J.: *The benefits of using BPMN and DMN together.* http://jtonedm.com/2014/12/04/the-benefits-of-using-bpmn-and-dmn-together/. Last accessed on 30.12.2017. 2.3

[Ti17] Tirelli, E.: *Building Business Applications with DMN and BPMN. Association for Computational Linguistics.* November 2017. 2.3

[Tu07] Turi, C., Ed.*Solvency II Framework.* Swiss Reinsurance Company. IUMI Conference. September 2007. 1.2.2

[UF96] Usama Fayyad, Gregory Piatetsky-Shapiro, P. S.: *The KDD Process for Extracting Useful Knowledge from Volumes of Data. COMMUNICATIONS OF THE ACM.* November 1996. 1.1, 2.1

[VP14] Victor Platon, A. C.: *Monte Carlo Method in risk analysis for investment projects. Institute of National Economy, Romanian Academy.* 2014. 1.2.3

[WC98] White, M.; Caldwell, T.: *EXEMPLARS: A Practical, Extensible Framework for Dynamic Text Generation.* pages 266–275. CoGenTex, Inc. 1998. 1

[Wia] Wikipedia: *List of best-selling books.* https://en.wikipedia.org/wiki/List_of_best-selling_books. Last accessed on 30.11.2017. 2.2.2

[Wib] Wikipedia: *Natural Language Generation.* https://en.wikipedia.org/wiki/Natural_language_generation. Last accessed on 30.06.2017. 1.1

[Wic] WikiPedia: *Non-functional Requirement.* https://en.wikipedia.org/wiki/Non-functional_requirement. Last accessed on 24.07.2017. 3.1.4

[Wid] WikiPedia: *Realization (linguistics).* https://en.wikipedia.org/wiki/Realization_(linguistics). Last accessed on 17.06.2017. 2

[Wie] Wikipedia: *Solvency II Directive 2009.* https://en.wikipedia.org/wiki/Solvency_II_Directive_2009. Last accessed on 30.06.2017. 1.2.2

[WNS92] Walker, M. A.; Nelson, A. L.; Stenson, P.: *A Case Study of Natural Language Customization: The Practical Effects of World Knowledge.* 1992. 2.1.3

[YS] YSEOP: *Official website of YSEOP.* https://yseop.com/resources/. Last accessed on 03.05.2017. 2.2, 2.2.1

[Yu04] Yu, J.; Reiter, E.; Hunter, J.; Mellish, C.: *Choosing the content of textual summaries of large time-series data sets. University of Aberdeen.* March 2004. 2.1.2