TAMPEREEN TEKNILLINEN YLIOPISTO
TAMPERE UNIVERSITY OF TECHNOLOGY

MOJTABA SAROOGHI

IMPLEMENTING A SELF SERVICE REPORT DESIGNER AND VIEWER APPLICATION

Master of Science Thesis

Examiners: Prof. Kari Systä, Prof. Kaisa Väänänen-Vainio-Mattila
Examiner and topic approved by the Faculty Council of the Faculty of Computing and Electrical Engineering on October 8th, 2014.

ABSTRACT

TAMPERE UNIVERSITY OF TECHNOLOGY
Degree Program in Information Technology
SAROOGHI, MOJTABA: Implementing a Self Service Report Designer and Viewer Application
Master of Science Thesis, 77 pages, 4 Appendix pages
August 2014
Major subject: Human Technology Interaction
Examiners: Prof. Kari Systä, Prof. Kaisa Väänänen-Vainio-Mattila
Keywords: Self Service Report Designer, Self Service Business Intelligence, Data Visualization, Data Interaction, Usability

Supporting users in dealing with huge amount of raw data and getting value out of data is one of the interesting and evolving topics in the IT world from both technical and usability points of view. The current extensive growth in the amount of data and significant consideration of data analysis in business decision making have forced companies to invest more on data management solutions and tools with innovative and user centered approaches.

This thesis work implements a self-service data reporting tool helping ordinary users to visualize, browse and analyze their data. This product allows *report designer* users to analyze and visualize their data and also design intractable dynamic reports for users who just want to see and play with ready reports. As a result, this product is consisted of two parts, one report designer and one report viewer application.

In the first part of this text some technical backgrounds considering report designer applications are mentioned, then the details of the development phase specific to the product under development and consisting of its design and implementation details are discussed and finally some evaluations of the product and design decisions are summarized.

Choosing development environment was a critical decision. The practical problems of selecting HTML/JavaScript as the main development technology is reported by development team. JavaScript codebase is hard to manage, there is not enough tooling support for JavaScript programming which reduces the speed of development, moreover HTML based pages which are hosted inside a web browser were not flexible enough to support users in interacting with UI (in report designer application),or in a better way, it is discussed that supporting users with a good experience in UI interactions for report designer application was a complicated task because of choosing HTML as UI technology. Furthermore, in the designed user's workflow model it was decided to empower users by providing them with a rich user control and freedom as the main feature of the application, though at the end, this design decision caused some complexities and lack of efficiency in the usability of the product.

PREFACE

This Master of Science Thesis is done in the Department of Electronics and Communications Engineering at Tampere University of Technology and Wapice Oy.

My special thanks to Professor Kari Systa and Professor Kaisa Väänänen-Vainio-Mattila for their support, expert guidance and valuable discussions during the supervision of my thesis and also during my study in TUT.

I would like to thank Wapice Oy management team for this opportunity to do my thesis there, my colleagues in Wapice and my friends in TUT for their help.

Finally, I give my sincere thanks to my family for their permanent support and love.

1 INTRODUCTION

> *"I keep saying the sexy job in the next ten years will be statisticians. People think I'm joking, but who would've guessed that computer engineers would've been the sexy job of the 1990s?* ***The ability to take data—to be able to understand it, to process it, to extract value from it, to visualize it, to communicate it—that's going to be a hugely important skill in the next decades****, …. Because now we really do have essentially free and ubiquitous data. So the complimentary scarce factor is the ability to understand that data and extract value from it."*

(Hal Varian, Professor of Information Sciences, Business, and Economics at the University of California at Berkeley / Chief Economist, Google)

It is a proven fact that data is a strategic asset in this contemporary world; there are numerous books and resources about the value of data and how companies should govern their asset (data) to be more successful in their business. The reason for the importance of data could be explained simply as companies (people) can use it to predict their future and make better decisions by the information they already have.

Moreover, nowadays it could be clearly seen the number of resources of data is increasing rapidly (sensors and signals everywhere, too many social websites and lots of online information, internet of things and…). According to a *Gartner* report done in 2010, enterprise data has expected to grow by 650 percent in five years [1]. As a result of increases in the amount of data, there should be some tools to help people get value out of large data sources. But the problem is with a small amount of data it is possible to manage data in some text files (even paper notes) or in case of a little bigger sources of data; spreadsheets and small data management applications beside legacy databases and report systems are sufficient, but when the data is coming from different sources and the amount of it is too much; companies (individuals) have lots of problems to cope with data in all different steps like gathering, integrating, retrieving, sharing, analyzing and most importantly in interpreting those data.

Data in its raw style is not interpretable by human. It is hardly possible to use data without any process because it is just some meaning less numbers, words (symbols or signs) and etc. raw and not ready to use by human. Processing data and giving it some meanings and contexts will bring data to its second level of maturity called information. Information by itself could be or could not be useful for making decisions, in this level still it is hard to get much benefits from data. The ability of finding relations between pieces of information, connecting and using them (interpreting information) to get practical benefits from it, is called knowledge. When people are empowered by knowledge they will have the ability to judge about the facts and they can make right decisions based on the current situation (current data). Wisdom is the reason why people put lots of efforts

to gain, store and process data. Formally wisdom could be defined as *"the ability to increase effectiveness. Wisdom adds value, which requires the mental function that we call judgment. The ethical and aesthetic values that this implies are inherent to the actor and are unique and personal* [2, pp. 5,6]*"*.The below flowchartish picture is trying to explain this journey of data from something raw and useless to something priceless and important which brings power for its owner. This graph is based on *DIKW Pyramid* theory (Data, Information, Knowledge, Wisdom) theory into account. [3]

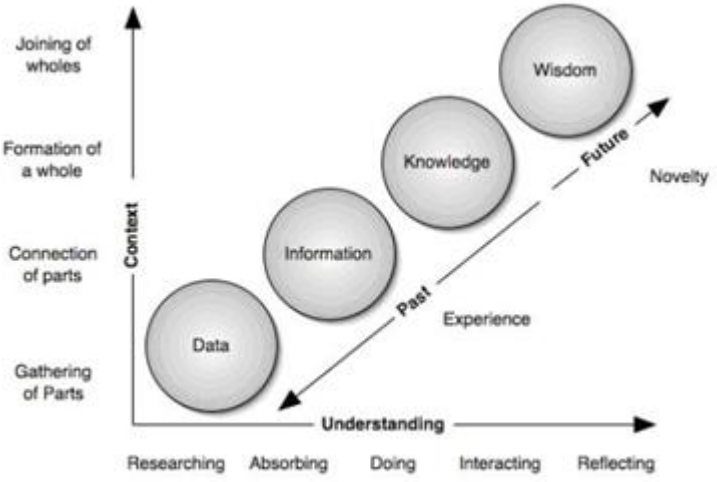

Figure 1 Journey of Data based on DIKW theory [4]

After an organizations accepted that their data is important, they should find a way to manage their data resources so they use some kinds of business analytics to get benefit from their data. In the next chapter terms like *Business Intelligence* and *Business Analytics* will be defined more precisely; but here generally it could be said that *BI* - including tools which is called *BI* tool - is a system used to transform raw data into meaningful and useful information to help businesses in decision-making and strategic planning process, *BI "encompasses all the capabilities required to turn data into intelligence"* [5, p. 3]. Regarding to DIKW theory it can be said that BI helps its users to gain knowledge and wisdom from their raw data in a less painful way.

Values of data and challenges in getting benefits from data, caused that analysis of data and business intelligence related product have an almost a fix position among top 10 technology trends in recent years. Industry studies have highlighted significant importance and developmental progress of data analytic systems and tools [6] [7]. Based on IBM technology trend survey on 4,000 information technology (IT) professionals from 93 countries and 25 industries; business analytics are identified as one of the four major technology trends in the 2010s [8]. Furthermore, according to a survey by Bloomberg Businessweek (2011), 97% of companies with revenues of more than $100 million are using some form of business analytics [9]. Taking the result of these researches, it is quite clear that this trend is emerging rapidly in the IT world (importance, usage and popularity of data analytics systems and tools).

The above paragraphs were trying to prove the importance of data for companies and the increasing trend in using business analytics among companies, though still there is a

fact; using data in a right way and getting benefits from data is a difficult task and it gets more complicated by growth in the amount of data and its variety. Bloomberg Businessweek (2011) stated that while businesses have warmed to the idea of fact-based decision-making, a still steep learning curve remains. *"Only one in four organizations believes its use of business analytics has been "very effective" in helping to make decisions* [9, p. 2]*."*

One important part of Business Intelligence systems or solutions are *BI* tools. These applications help users to interactively browse (mine) their data to find and understand relations and patterns inside their data and get insight of it (including designing and viewing data reports). A short look at the market of BI tools and different surveys (like the one mentioned in the previous paragraph) reveals lack of a good usability in current BI tools and a niche in which giant software companies are competing to fulfil by their (human centered)designed BI tool. Usually users of BI tools (in the front end UI level) are people with different technical knowledge, so a successful tool should support all the different users in an engaging, easy to use environment.

SSBI (**Self-Service Business Intelligence**) is a kind of **DIY** (**BI**) (**Do It Yourself Business Intelligence**) emphasizes more on ordinary (not something just for IT professionals) users of BI. The focus is on making end users free of needing help from IT staff while they are interacting with BI tool. Claudia Imhoff and Colin White in their research paper defined SSBI as *"The facilities within the BI environment that enable BI users to become more self-reliant and less dependent on the IT organization. These facilities focus on four main objectives: easier access to source data for reporting and analysis, easier and improved support for data analysis features, faster deployment options such as appliances and cloud computing, and simpler, customizable, and collaborative end-user interfaces* [10, p. 5]*."*

It is worth pointing that developing a BI/SSBI application needs a big team and lots of resources and it takes years to finish a full featured BI/SSBI, it has a long list of functional and usability requirements to fulfil. This thesis work is based on a development process to design and develop a small part of a SSBI tool (or maybe in this level an ad hoc report designer).

This project is based on real needs of ***Wapice's*** customers. Our customers have had difficulty to adopt to current BI applications in the market made using current BI/SSBI applications not as efficient as they expected, so from ***Wapice*** management team it was recognized as a potential niche to take by a customized (based on real customers' needs), small scale and simple version of a reporting tool which caused this product to be born. It was planned to have a prototype version of the application in the first phase of development process to reduce the risk of heavy investment and start a big application from the ground without analyzing it in depth. This thesis is a report based on development of this product.

In this first version the goal was providing an executable prototype application to be able to evaluate technical and usability decisions for future more mature versions, so there was not a big investment on the project's resources and development team was small the

whole process took something like one year. It was tried to limit the scope of the application and focus on the most important requirements and features and also it was tried to have a working implementation of those features to able to have a robust realistic estimation about: first the possibility to develop a mature and competitive SSBI application by company (to find a realistic idea about the amount of investment on time and human resources needed for this complicated project), second to be able to evaluate technical and usability decision decisions in a short version of application (before starting the main project) and finally having a working prototype to be able to present and sell the main application idea to customers.

In summary, the purpose of this thesis work is designing and implementing a part of SSBI tool (report designer and report viewer) and outlining the technical and usability difficulties and issues of the implementation.

In the next chapter some literatures about SSBI/BI and Human Computer Interaction (*HCI*) are discussed, then the third chapter talks about BI/SSBI market, user groups and requirements generally and in the fourth chapter the focus is on the product under development and its design decisions, chapter five contains technical information about the application and finally in the sixth chapter some evaluation about technical and usability of the product is described.

2 BACKGROUND

Data exists so the owners of data (organizations or people) need some systems and tools to be able to manage and get benefits out of data. These systems and tools are composed of many components to facilitate various operations in different levels of data processing from gathering data, storing and structuring data to designing and generating productive reports of data. In this section the developmental progress of these kinds of tools and systems from technical and usability point of view also some concepts related to Human Computer Interaction which helps to produce usable products is discussed.

2.1 Reporting Systems

In the past reporting system was a part of the integrated information system and they were using the same database as operational information system (which they were a part of it). At that time in business/financial data information software, reporting systems were sharing the *OLTP* storage with information systems. An *OLTP* (On-line Transaction Processing) stores and handles transactions of information systems, they handle a large number of short on-line transactions (INSERT, UPDATE, DELETE). In an OLTP the focus is on supporting query processing with a high speed, and keeping data integrity in a concurrent environment. These kinds of design concerns in OLTP data storages put many limitations on reporting systems which are expectable because OLTPs are not designed in a way to support reporting inherently and reporting has the second priority in these systems. Figure 2 shows the structure of these kinds of systems [11].

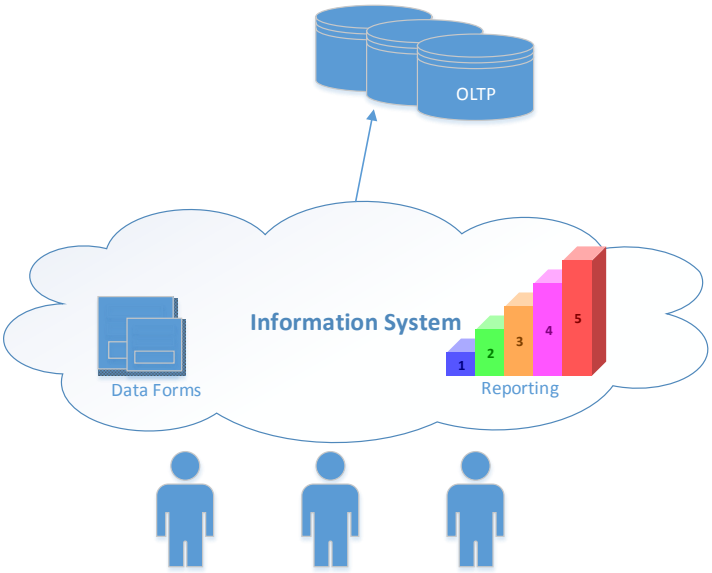

Figure 2 Legacy reporting system as a part of information system sharing operational OLTP as its storage

The growth in the amount of data, beside the growth in data processing requirements caused dedicated systems for reporting purpose come alive. In this generation of infor-

mation systems some concepts like *Data Mining* and having a dedicated storage for reporting purposes - *Data Warehouse* - got popular and there was a clear movement from operational databases to data warehouse history storages and *OLAP* (On-line Analytical Processing) systems. A data warehouse is the heart of a reporting system in modern data processing/analyzing systems. Usually a data warehouse stores the business history data of an organization collected from various sources such as online transaction processing (OLTP) systems, legacy systems, text files, invoices, emails, spreadsheets and etc. Data in a data warehouse is stored in a way to support faster and easier data analysis rather than to handle real-time transactions [12].

OLAP (On-line Analytical Processing) part of a data warehouse has a responsibility to allow data warehouses to be used effectively for online analysis and providing rapid responses to complex analytical queries. OLAP applications are widely used for Data Mining purposes. In OLAP systems there are some tools and techniques (like cubes, multidimensional data models and data aggregations) enabling reporting systems to answer to queries of historical data rapidly. OLAP systems give people the ability to analysis data and help them to make right decision by mining and retrieving patterns from their old data [13].

Technically OLAP and Data Warehouse are two separate components in a data reporting system but for the sake of simplicity these words can be used interchangeably. Figure 3 is revealing the structure of these new data processing and reporting systems.

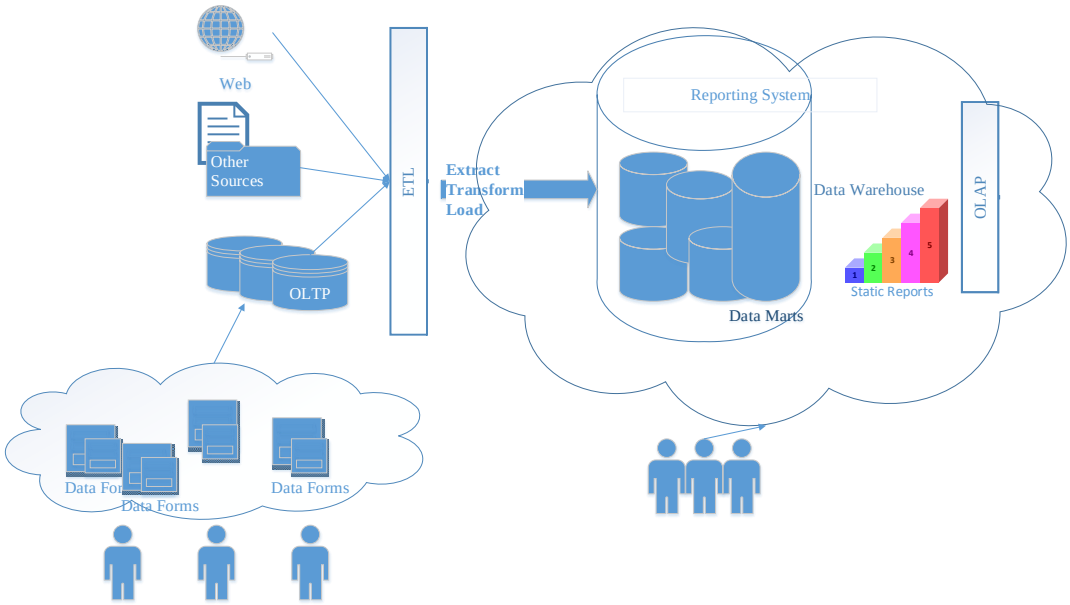

Figure 3 New reporting systems' structure in presence of data Warehouse and OLAP

2.2 Analyzing Data

Data processing systems should be more helpful than just generating reports and representing data, the next generation of data processing tools were trying to help people with data analyzing process not just with facilitating reporting needs but also in data mining and decision making.

Business Intelligence - a modern term used to describe a system composed of different components, tools and players supporting businesses in making right decisions based on data they have gathered - is the result of developmental progress in a series of related attempts to facilitate data analyzing for human. Two related concepts close to *BI* are Data Mining and Decision Support Systems (DSS).

Data Mining or *Knowledge Discovery* is a process of retrieving important data from a bunch of related data. Data mining tools enable their users to discover various patterns, generalizations, regularities and rules in data resources [14]. Fayyad and others define data mining as *"the application of specific algorithms for extracting patterns from data* [15, p. 39]*"*. It can be said the overall goal of the data mining process is to extract information from a data set and transform it into an understandable structure for later use. Data Mining has a broad usage in different fields like, science and engineering, medical, statistics, business and etc. Also there are different related concepts to data mining like text mining, music mining, web mining, image mining and etc. Data mining from algorithmic point of view could be helpful in every data analyzing tool (like a BI tool) to guide users to their answers (probably with some machine learning related algorithms). Besides, data mining from the idea of mining data to get valuable result is exactly one of the most important services of a data analyzing tool which allows its users to mine their data and get insight in their data.

The definitions for design support systems (*DSS)* are various based on the authors' viewpoints. *Finlay* and others define a DSS broadly as *"a computer-based system that aids the process of decision making* [16]*"*. Also according to Turban, DSS is *"an interactive, flexible, and adaptable computer- based information system, especially developed for supporting the solution of a non-structured management problem for improved decision making. It utilizes data, provides an easy-to-use interface, and allows for the decision maker's own insights* [17]*"*. A DSS helps its users in making decisions in different situation by answering users' questions. To make it clearer we can refer to a sample of DSS system by Power:*"... a manager might query a database to ask questions like what is total sales for each of the last five years; what items have been out of stock more than 5 days in a month; and which customers had the most orders ($ value) in 1996. Managers may also ask questions like are we meeting profit targets; and what salespersons are meeting their sales goals ...* [18]*"*and the DSS will try to answer these questions by rules and data it has in its storage (probably with help of a rule engine).

From a historical point of view seems BI is originating from DSS. Based on Wikipedia *"Business intelligence as it is understood today is said to have evolved from the decision support systems (DSS) that began in the 1960s and developed throughout the mid-*

1980s. DSS originated in the computer-aided models created to assist with decision making and planning [19]*."*

Like the other terms discussed earlier, it is hard to find a unique and precise definition for Business Intelligence (BI), *Parr Rud* says that Business Intelligence is *"a term that encompasses all the capabilities required to turn data into intelligence* [5, p. 3]*."* This definition is a broad definition of BI process pointing the main purpose of BI transforming raw data to something useful and valuable (intelligence). *Gartner (2011)* defined BI as *"an umbrella term that spans the people, processes and applications/tools to organize information, enable access to it and analyze it to improve decisions and manage performance* [20, p. 15]*."* ***Sabherwal*** defines it *"the process of providing decision makers with valuable information and knowledge by leveraging a variety of sources of data as well as structured and unstructured information* [21, p. 5]*."* Forrester has two definitions for BI a broad definition of BI *as "Business Intelligence is a set of methodologies, processes, architectures, and technologies that transform raw data into meaningful and useful information used to enable more effective strategic, tactical, and operational insights and decision-making* [22, p. 1]*."* Also Forrester has a newer definition of BI which is narrower *"... referring to just the top layers of the BI architectural stack such as reporting, analytics and dashboards* [23]*."*

Sabherwal in his brilliant book (*Business intelligence : practices, technologies, and management*) explains that Business intelligence term is used both for the process through which organizations or people obtain useful business knowledge from their raw data, and also BI is used for the result of data analyzing process meaning the insight ones gets from analyzing data. Besides, it is worth noting that there is a difference between a BI tool developed by software companies and a BI solution developed within an organization. An organization inside itself develops a BI solution utilizing one or some BI tools to reach to their goal which is getting insight in their raw data. So there are four BI related terms *BI Product, BI Process, BI Solution and BI Tools* in Figure 4 relation between these terms are shown. [21]

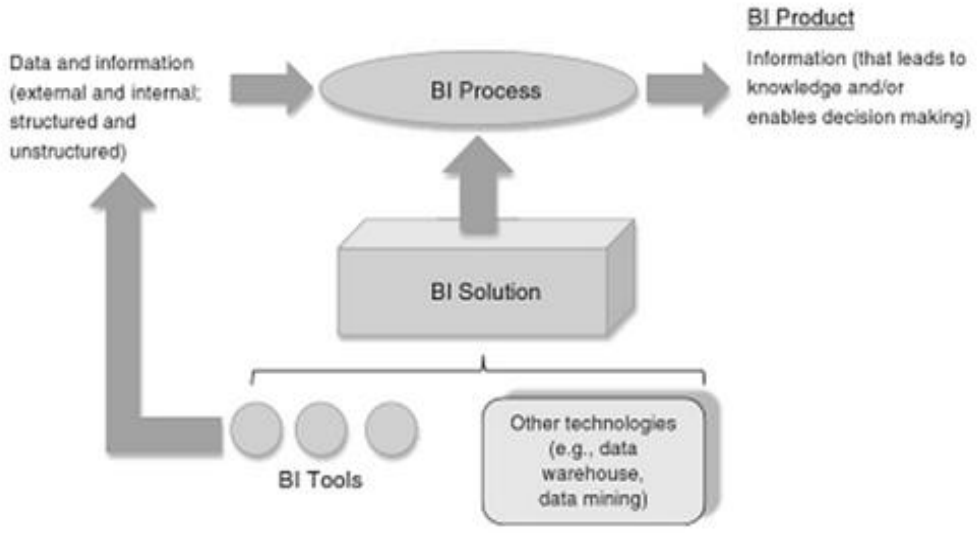

Figure 4 BI Product, BI Process, BI Solution and BI Tools [21, p. 6]

Finally, another term quite related to BI emerging recently is *Business Analytic* or *BA*, BI and BA concepts have almost similar meaning though some authors claim that BI is a part of BA and some claim vice versa, for the sake of simplicity in this report these two terms are used interchangeably.

2.3 Human Computer Interaction (HCI) and Human Data Interaction (HDI)

Human Computer Interaction (HCI) is a wide term for all topics studying computer and software application in presence of human. A significant number of academic institutions now are studying HCI to increase the usability and support good user experience for humans using computer related technologies.

Association for Computing Machinery (ACM) defines human-computer interaction as *"Human-computer interaction is a discipline concerned with the design, evaluation and implementation of interactive computing systems for human use and with the study of major phenomena surrounding them* [24]*."*

A dedicated topic under HCI studies design and implementation of interaction techniques between human and computer in particular. Interaction techniques in this level are defined as combination of software and hardware (input/output devices) which help humans to communicate with computers (give inputs to and get results from computers).

Particularly in this context, *Tufte* has defined this interaction as two powerful information processors (human and computer) attempting to communicate with each other via a narrow-bandwidth, highly constrained interface. [25]

Technically, an interaction technique involves 3 components:

- One or several input devices that capture user input
- One or several output devices that display user feedback
- A piece of software that: interprets user input into commands the computer can understand, produces user feedback based on user input and the system's state.

Although human interaction with data could be seen as a part of HCI, when data is the main matter; it is relevant to discuss human interaction with data in more detail. Because of the growth in the presence of data in everyday human life and its value, recently a new topic in IT world is emerging quite fast is called Human-Data Interaction (HDI). HDI generally is studying human in presence of data to provide a good experience for human in interacting with data. Technically because HDI is quite new term in IT there is not an accurate definition of HDI or even not an agreement about the coverage of HDI in different definitions. Niklas defines HDI as *"the human manipulation, analysis, and sense making of data* [26, p. 1]" which is quite limited to data visualization. There are some attempts to define a model (Figure 5 shows this proposed model) for HDI and clarify this topic suggesting a border definition for HDI, this new definition is related to HCI definition but focused on acquiring, analyzing and sharing data more than generic interaction with computer or user interface implying *"HDI overlaps HCI but is not contained within it* [27, p. 5]*."* [28]

This new definition for HDI is aiming to cover all different aspects of human data interaction in an environment with both users and data. Based on this definition HDI could cover the whole process of data interaction in all levels between human and data from gathering to analyzing and getting benefit from data, also sharing, visualizing and interacting with data (UI level), privacy and security of, even data processing and machine learning algorithms, and etc. Figure 5 shows this new model for HDI. [27] [28]

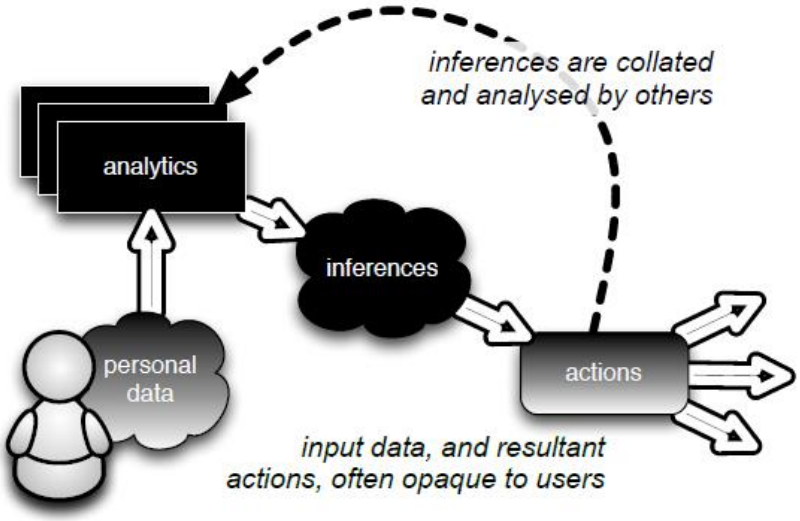

Figure 5 HDI broad definition diagram [27, p. 5] [28]

The ubiquities of data and data driven applications in human day to day life is expected to cause a significant progress in HDI field in near future. Still HDI is in its early stage and there are not many researches and resources for this new field. HDI first should answer to lots of questions like: How applications should supports users dealing with data around themselves? What is a best data exploration experience? How applications should help their users to understand data and get benefits of data? How applications should present data to their users and how users can interact with presented data in a convenient way? How users can ask questions and how applications should be fast in answering those questions? What human factors are influencing on users experience dealing with data? If data is everywhere how an application should supports its users' privacy? What kind of data sharing interactions are better than others? And lots of more questions which HDI is supposed to answer to be able to help developers in providing a good user experience while human dealing with data in its life.

2.4 Data Visualization

Any tool that supports human in understanding data and relation between facts has to deal with visualization to make abstract concepts concrete for human. Everybody has the experience of understanding a complex concept (needing lots of explanations) just by looking at a well-designed picture (visualization) and that is why people are believed in "*one picture is worth a thousand words*" by heart. The reason behind importance of data visu-

alization comes from the limitation of human cognition and memory. *"Well-designed visual representations can replace cognitive calculations with simple perceptual inferences and improve comprehension, memory, and decision making* [29, p. 59]*."*

Data in its lowest level - stored in data storages like databases, binary files - is in machine understandable format (like binary) and not usable for human. By processing binary data and formatting it, people can use text files or text representation of data record in tables or word processing applications, even in this level presenting textual data to users in a right format increases the speed of comprehension of data; for example adding a currency sign to the price of items in a bill helps people to have a right mind set of what those numbers stand for in the bill. In the next levels data could be shown by different rich representations to enhance and engage users' perception using graphs, charts, images, other media (videos, sounds, animates) and etc.

Data visualization is a broad and multi-disciplinary topic; a contribution of human psychology, computer science, cartography, art, etc. investigating firstly how human visual perception system works **(**how people use visualizations to gain insight) and secondly design visualization techniques to help people get best from data.

Data visualization is the presentation of data in a graphical format to help people find and understand the concept behind data faster and easier. Technically Computer-based visualization can be defined as: *"... the use of computer-supported, interactive, visual representations of data to amplify cognition* [30, p. 6] *."*

Beside technical aspects of a good visualization; the challenging part is creating *entertaining* and *engaging* visualizations based on the type of data and different contexts and users. Following the principle "*attractive things work better* [31]" innovative and aesthetic design for visualizations are key factors.

It must be pointed that there are two different kinds of data visualizations: scientific visualization and information visualization. Information visualization tries to visualize *semantics,* or meanings of information and in contrast to *scientific* visualization, information visualization deals with nonnumeric, non-spatial data [32]. *S*imply information visualization usually is used with nonscientific data for non-scientific usage, for example graphical representations of data for businesses.

Business analytics deal with business data so data visualization techniques which are used in BI tools are mainly information visualization techniques (not scientific ones). By a short look at the current BI tools in the market it can be seen there are different popular and standard techniques for visualizing data: different kinds of charts (line, bar, pie, column, scatter, area etc.), maps, 3d maps, heat maps, data tables, matrixes, custom gadgets, video and etc. but still lots of BI vendors are trying to find more innovative and engaging ways to visualize data in their BI tools because BI companies already know beside data interaction techniques, data visualization techniques could be a key factor in success of a BI tool.

2.5 Usability

The goal of producing a product is providing something useful for users, but what is a useful product? A product should fulfil users' needs, it should do what users need (it should have utility attribute). Besides supporting functional requirements a product should be usable for its users. Informally a usable product can be defined as an easy to use and easy to learn product. Based on ISO 9241 usability is "the extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use. " [33]
Effectiveness, efficiency and satisfaction mentioned in the above definition could be described as below [34]: "

- **Effectiveness**: the accuracy and completeness with which specified users can achieve specified goals in particular environments
- **Efficiency**: the resources expended in relation to the accuracy and completeness of goals achieved
- **Satisfaction**: the comfort and acceptability of the work system to its users and other people affected by its use"

Jakob Nielson defines usability more practically as *"usability is a quality attribute that assesses how easy user interfaces are to use* [35]*."* He defined usability as below 5 components [35]:

- **"Learnability**: How easy is it for users to accomplish basic tasks the first time they encounter the design?
- **Efficiency**: Once users have learned the design, how quickly can they perform tasks?
- **Memorability**: When users return to the design after a period of not using it, how easily can they reestablish proficiency?
- **Errors**: How many errors do users make, how severe are these errors, and how easily can they recover from the errors?
- **Satisfaction**: How pleasant is it to use the design?"

In summary useful, usability and utility feature of a product can be defined as below [35]:

- **Utility** = whether it provides the features you need.
- **Usability** = how easy & pleasant these features are to use.
- **Useful** = usability + utility.

It can be said a useful product has both utility and usability attributes and the importance of these two attributes are almost equal. Users can use a useful product to do what they want and they can do that in an easy way. [35]

Usability of a product could be evaluated in 3 ways in each a different source is used to evaluate usability of a product: users, usability experts, or models could be used as the main source to evaluate a product. [36]

2.5.1 User-based Evaluation (Usability Testing)

Usability testing refers to evaluating the usability of a product by testing it with users. Typically, during a test, participants will try to complete some tasks while observers watch, listen, takes notes and capture them. The goal of this process is identifying any usability problems, collect qualitative and quantitative data and determine the participant's satisfaction with the product. Actually in this kind of test the goal is measuring effectiveness and efficiency of users work, and the amount of user satisfaction while users are using the product in a testing environment. [37]

2.5.2 Expert-based Evaluations (Usability inspection)

Expert evaluations of usability are similar to code walkthroughs and design inspection in software development. Experts use different evaluation methods including heuristic evaluation, guideline reviews, pluralistic walkthroughs, consistency inspections, standards inspections, cognitive walkthroughs, formal usability inspections, feature inspections and etc. to inspect usability of a product. Although these methods do not involve users, they can provide useful insights about usability of a product. [36]

In the area of software user interface heuristic evaluation technique is the most widely used inspection method. Heuristic evaluation uses some evaluators to assess how much an interface complies usability design principles. There is a list of ten evaluators popular for evaluation user interface provided by Jakob Nielsen broadly used in user interface evaluation which can be used as a reference for heuristic evaluation.

2.5.3 Model-based Evaluations

Model-based evaluation uses a model of how a human interact with the proposed system to measure usability problems of the system replacing user testing in a repeatable, cheaper and faster way. The same way as models are used in other engineering disciplines, engineering models for usability can be used to predicate how well users can perform tasks using the product under development. Model-based evaluation can be used to get some usability results even before implementing a prototype or testing with users just by evaluating a human-computer interaction model called engineering models or analytic models for usability. For some sample of usability models can refer to famous models like task network models, cognitive architecture models, GOMS models and etc. [38] [39, p. 58]

David Kieras explains these models in a good way as *"The model is based on a detailed description of the proposed design and a detailed task analysis; it explains how the users will accomplish the tasks by interacting with the proposed interface, and uses psychological theory and parametric data to generate the predicted usability metrics. Once the model is built, the usability predictions can be quickly and easily obtained by calculation or by running a simulation. Moreover, the implications of variations on the design can be quickly explored by making the corresponding changes in the model* [39, p. 58]*."*

2.6 Self Service Business Intelligence (SSBI)

"IT Can No Longer support The Majority of Bi Requirements BI is unlike any other enterprise software in that it exhibits a paradox: The more BI requirements you implement, the more new ones pile up. Relying too heavily on IT-centric BI support models is not sustainable [40, p. 1]*."*

"*The facilities within the BI environment that enable BI users to become more self-reliant and less dependent on the IT organization. These facilities focus on four main objectives: easier access to source data for reporting and analysis, easier and improved support for data analysis features, faster deployment options such as appliances and cloud computing, and simpler, customizable, and collaborative end-user interfaces* [10, p. 5]*."*

As it was described above BI as a whole Eco systematic process and solution has a layered architecture with different components, in each layer there are some different tools. At the front end in a BI solution usually there are some BI tools allowing users to mine their data, design their reports, view and filter data interactively. The users of this UI tools come from different background and have a various range of technical knowledge, so a successful BI tool should support all different users in an engaging way within an easy to use environment.

The comprehensive investment and focus to fulfill the requirement of a BI tool to support ordinary users with different backgrounds by a good usability and user experience caused the term Self Service Business Intelligence (SSBI) comes to alive recently. SSBI is an approach to data analyzing that focuses more on business users and tries to enable them to access and work with business data without involvement of organization's IT department.

Generally SSBI empowers business users to handle their own data analyzing needs without bureaucratic, time and resource consuming dependencies to IT staff. It means business users would be able to analyze data themselves and do not need to ask IT staff to design and bring them some readymade reports. In fast forwarding businesses world, the ability to gather and analyze data quickly is a big help for companies. Imagine a segment manager of a small branch of a big company at the end of the year wants to have some understanding about the amount of selling and the profits of the branch under his control. Big companies have a central IT department, so the segment manager should ask for some reports based on some criteria, then one IT guy (hopefully experienced not just in IT also have a good understanding of the business as well) will design some reports based on the requirement and send it back to the manager. This bureaucratic processes to make this request and answer happen need some time, besides because communication mistakes and barriers the manager might not be satisfied with reports so he will ask again for some new versions and the cycles goes. After some days (or some months), finally the manager is satisfied with reports so he studies the result and will find some interesting patterns in finical data from reports. The next step for the manger is going to details of

those patterns and data but that data is not available in those ordered reports so the request, answer and wasting time will start again from the beginning. With the presence of SSBI the segment manager can easily connect to the data storage (with settled permissions) and then he can design and mine business data himself without involvement of IT people, which reduces lots of communication costs in both money and time.

Formally SSBI has defined as "*The facilities within the BI environment that enable BI users to become more self-reliant and less dependent on the IT organization* [10, p. 5]. "

Those facilities focus on four below goals:

- Make BI Results Easy to Consume and Enhance
- Make BI Tools Easy to Use
- Make Data Warehouse Solutions Fast to Deploy and Easy to Manage
- Make Data Sources Easy to Access

Figure 6 shows the four objectives of SSBI.

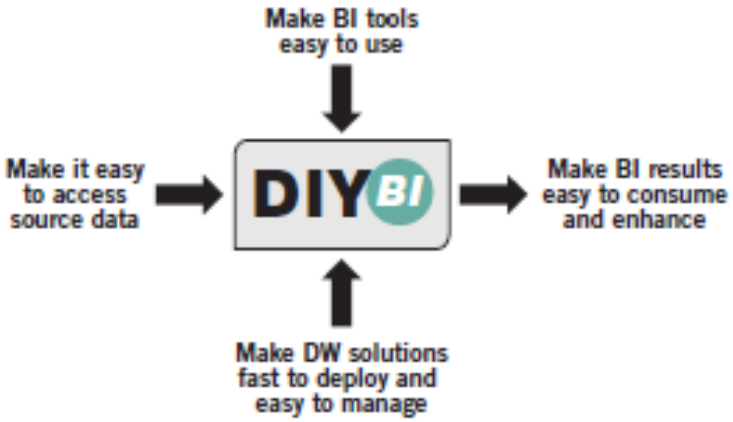

Figure 6 the four objectives of do-it-yourself or self-service BI [10, p. 6]

In summary, the spirit of SSBI (providing good user experience for business users while they are interacting with data) indicates the importance of studying usability concepts beside technical ones in designing a successful BI tool. The purpose of this study is an attempt of designing and implementing a part of self-service BI (self-service report designer and viewer) tool and reporting the technical and usability issues. In this research the focus is mainly on data interaction and data visualization in user interface level of SSBI.

3 MARKET, USERS AND REQUIREMENTS

The first step of developing any product is understanding users, users' requirement and future product market (more generally product context). Development team should have a clear mindset about what they are going to develop, which group of users they are targeting, which requirements they want to fulfil and generally how the context of product looks like. There are lots of resources about how teams should perform a software (product) development in user centered style. For example in *http://www.usabilitynet.org/* different developmental activities for each step of software process based on the amount of a project's resources are listed. In this method of software development for the beginning phases which are *planning & feasibility* and *requirements gathering* the emphasis is on customer meeting, users, context and market study (activities like stakeholder meeting, competitor analysis, user surveys, interviews, contextual inquiry, user observation, scenario of use and etc.). [41]

Unfortunately because the scope of this project was quite wide and the lack of resources and time it was not possible to conduct a detailed requirement study for this thesis work. Here in this chapter some short and generic requirement study of BI and SSBI applications **based on online resources** and also based on the project manager and developers' experience and understanding of BI and SSBI tools, are discussed (which could give a good understanding of the product underdevelopment, but not much precise and complete. A complete requirement gathering study for a big project topic like SSBI could take several months and needs a full team and can itself be a topic of a thesis).

3.1 An Overlook to BI and SSBI Market

In the current BI/SSBI market there are many good applications from giant software companies like Microsoft, IBM, SAP and etc. implying a big need of a detailed market review and analysis which itself could be an important part of requirement study. One of the best resources to get deep information of current vendors of BI/SSBI and successful applications in this market is *Magic Quadrants for Business Intelligence and Analytics Platforms* report published by *Gartner*. *Gartner* is one of the most famous and reliable technology research and advisory companies headquartered in Stamford, Connecticut, United States. They produce comprehensive collection of analyses, advice and researches for clients and vendors of different technologies to help them in understanding current technology world. [42]

Magic Quadrants research has been being published each year for quite long times. It positions and reveals technology vendors within a specific technology market. In these reports venders are divided into four groups *Leaders*, *Visionaries*, *Challengers* and *Niche Players*. These research papers (for each technology or field) reveal the *leaders* in a specific technology meaning venders who are executing well in the current market of that

technology and have a good vision for tomorrow as well, the *visionaries* who are the vendors having a good vision for the future but do not yet execute well enough in the current market, also *challengers* companies who are doing well today and dominating a large segment currently, but do not seem to have a good vision for future , and finally *niche players* are companies who are performing successfully on a small segment, but not wildly successful currently and not seems to have a good vision for the future. By analyzing these research papers and reports; ones can have a good view of almost all the big players in the market and understand about what is working now and what is going to work in the future for a specific technology. [43]

The result of Magic Quadrants for BI research is based on combination of a survey of customers conducted in *October 2013* with 1,935 company respondents and some detailed RFP questionnaires by vendors. Figure 7 is based on this report and shows the players in this field. The right top square is showing leaders which are the main players. As it is visible *Tableau, Qlik, Microsoft, SAS, SAP, IBM Cognos, Tibco, Microstartegy and Information Builder* are the most important vendors in this field. [44]

Figure 7 Magic Quadrant for Business Intelligence [44]

Furthermore Figure 8 from a research by *Forrester* in 2012 which is dedicated to SSBI, again showing that *IBM, Microsoft, SAS and SAP* beside other companies mentioned in the above are the leaders of SSBI market, revealing that almost those companies which have a strong BI tool also are leaders in the SSBI market. [40]

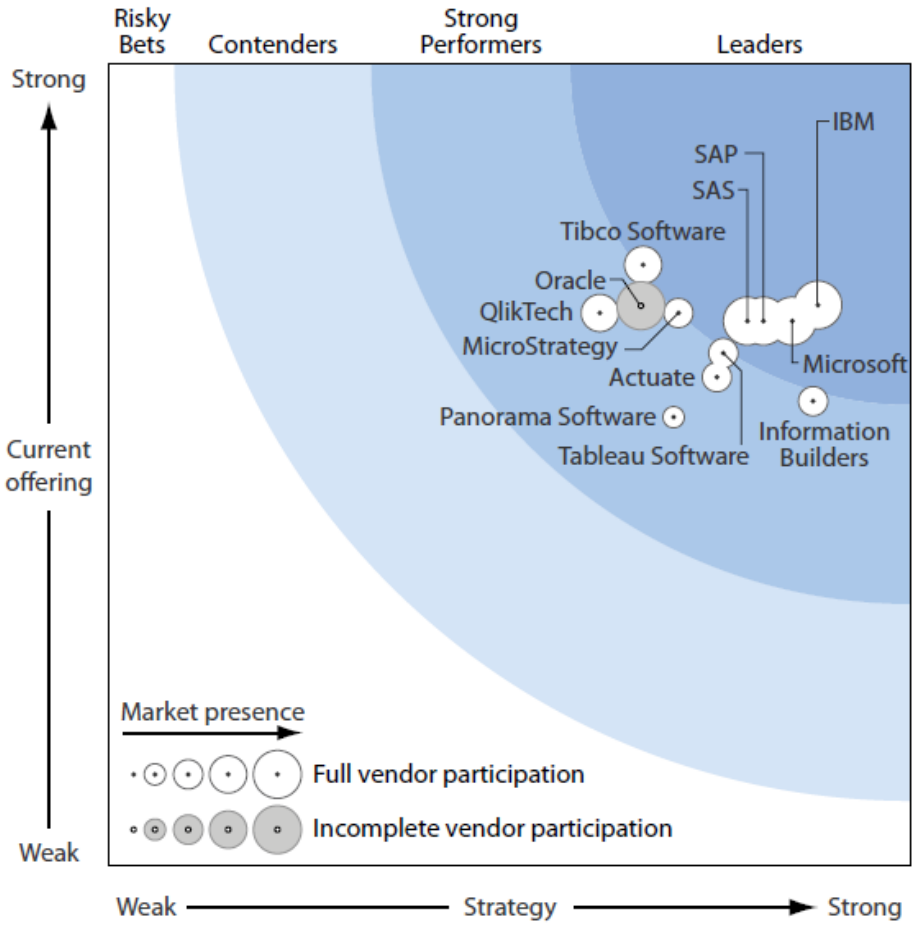

Figure 8 Forrester Wave™: Self-Service Business Intelligence Platforms [40]

When the project was started the team had a short look at some BI applications (from Microsoft, QlikTech, Tableau, MicroStrategy,..) it was easily observable these big companies are investing heavily on researches to improve their product with new innovative features (in both technical and usability) to be able to dominate SSBI market in near future and it was clearly visible there is a big competition between big vendors in this market. The focus from technical part is more on *MobileBI, Embedded BI and HTML based* environment and from usability point of view is on inventing some new engaging and easy to use methods to allow users to find, model and draw visualization of their data.

3.2 Users, User Groups and Personas

To find users' requirements for a product, it is important to know the users of the product first. Clarifying and classifying users into user groups help production team to have a concise and precise definition of the users of the product. A user group can be defined as a group of people who have similar interests, (probably) the same level of technical knowledge, usage goals, and concerns about the product which makes them to have the same requirements and expectation from the product. Understanding different user groups help developing a right product which covers all users with different requirements.

Users of a BI tool coming from a wide variety background and have different expectations from the application. Based on *Boris Evelson* BI's users groups can be classified in 5 classes as below: [45]

- **Casual Users** - Business people who consume reports or dashboards infrequently (typically less than weekly).
- **Executives** - Business people, managers, or decision makers who consume structured reports or dashboards to make daily decision based on data to drive their business inside their company quite regularly.
- **Average, Regular Users** - Business people who consume reports or dashboards frequently (typically each week) they are more independent than **Executive** in using BI tools on relying to IT department.
- **Power Report Users (**including **Analytic Experts)** they are business experts in analytic techniques and related toolsets, they need the least help from IT department in using BI tool.
- **BI Application Developers** – This group is including both people experts in developing presentation layer that support data access and report designing also data experts who are good at ETL, data toolsets and methods which both are part of IT department or production team.

In Table 1 these 5 user groups, their knowledge of IT (IT reliance level), their expectation from BI and their relation to the data analysis (producing or consuming of data) are shown [45].

Table 1 BI tools user groups [45]

	Consumers of Info	*Producer of Info*	*Data Interaction Type*	*Rely on IT*
Executives	✓✓		Reports Dashboards	✓✓✓
Power Users	✓✓✓	✓✓✓	Queries OLAP	✓
Average Users	✓✓	✓	Reports Queries OLAP	✓✓
Casual Users	✓		Reports Dashboards	✓✓✓
Developers	Support	Support Produce	Support	NA

These four columns are showing below information about each users group:

- It shows each group is consuming or producing (or both) information and how important this requirement is for that group. "What's interesting is that all our research shows is that most of the BI personas now are both consumers and producers of info [45]."
- It is revealing the type of data interaction techniques which is used in each group(queries, dashboards and OLAP)
- And finally showing the level of dependency on IT for each group

In this work the focus is on report designers which are part of **power users** and **average users** and report viewers (end users) which are part of **executives** and **casual** users group. In the next section some sample personas for users of these user groups are shown. Executing a real user or contextual study was out of the scope of this work so personas mentioned in Figure 9, Figure 10, Figure 11 are based on the informal information (from team experience) and some stereotypical personas for a BI tool retrieved from online sources (mostly from *Self-Service Business Intelligence: A Workbook* published by *Jaspersoft Corporation* [46]) they are mentioned here as a sample of real personas for a BI/SSBI tool.

Big Boss Billy

Age	50
Job Title	VIP Operation
IT Experience	Experienced
User Group	Casual Users

BACKGROUND

Bill is kind of a big boss and he makes big strategic decisions to make sure that the company is running as should it go. He is responsible and concerned with the entire company operation, including sales, production, distribution and support. He is quite good using IT technology and internet, and he believes in investing money on IT drives his business to success.

MOTIVATION

Bill wants to be able to get the most important information from data which they are gathering by different scenarios in the company. He wants to get enough information quickly to be sure company is on the track and also to react based on the situation. He does not have enough time to go inside of data, usually he asks the company's data analyzers to prepare him some data reports and mainly he get presentations of those reports by his employees. Beside those presentations he likes to have some interactive report that himself will be able to play with them.

FRUSTRATIONS

Bill sometimes gets a bunch of reports, but usually it is hard and time-consuming for him to find exact information which he really needs from those reports, because usually these reports are detailed in some parts (which he (probably) does not need them) and some parts (he really needs) are not even mentioned. So he decided to hire some business analyzers, but still sometimes he needs to make his hands dirty with reports and data.

IDEAL EXPERIENCE

Bill wants intractable, semi-automatic and visualized rich reports that he could filter and drill down charts and tables when he needs more details. He does not want to do a lot of analyses or learn complicated and advanced data interactions of complex reporting tools. He knows how to read charts and get main points from diagrams so he needs a tool to support him dealing with graphs in an intractable and easy to use manner to get immediate and most important information of data. He likes to get information really fast, and he is interested in new visualization like geo maps and animated graphs, his motto about ideal tools is "short learning curve, simple to use but rich insight".

Figure 9 Casual User Persona

Product Owner Joe

Age	37
Job Title	Product Owner
IT Experience	Advanced
User Group	Average Users

BACKGROUND

Joe is a business executive and product owner in their company. He should convince his clients about the company's products and schedules, keep track of the project with production team in the company, at the same time he reports the progress to the management team in the company. He is an advanced IT user quite good with excel and power point even he has some programming knowledge which helps him to write micros in excel to do some customization.

MOTIVATION

Joe is usually presenting data, metrics and schedules at weekly meetings inside the company for production teams and outside of the company in sessions with customers also in some monthly meetings with management team. He usually gets some reports from IT department but based on his idea and experience they are not sexy and rich enough to be presented in different occasions for different kinds of users, also the waiting period to get new reports from IT department is long so he decided to produce his reports by himself. He asked for some raw data from IT department and then tried to build his own reports in excel. Because lots his reports are almost with the same structure he is thinking of designing some primitive application to automate report generation process but he does not have that much technical knowledge to do that.

FRUSTRATIONS

Joe does not like work close with IT department in his idea technical guys usually do not think about the importance of a sleek UI, they are mostly into accuracy and they do not have marketing concerns. Besides, he needs his reports and personation really fast so it is not applicable to depend on IT department for weekly presentations. There are some reporting tool sets in the company but usually they are not working or the result are rough for presentation. And because of those problems, he usually uses excel and make his own reports.

IDEAL EXPERIENCE

Joe wants to work with fresh data, and have his customized reports with sleek visualizations, he wants to be able to export chart and other visualizations to presentable formats (like Power Point) and he needs freedom in designing visualizations. The most important point for him is having a designer which allow him to work with data without dealing with IT department too much. He wants to be able to create his own reports and visualizations to be able to answer his customers and mangers questions, but without having to know a lot about the data, data analysis concepts and database structures. Also the ability to share, search and reuse his (or other colleagues' reports) can help him to be as agile as possible. Finally he is really interested in new ways of data interactions and fancy dashboards to be able to get the attentions of his customers as much as possible. His motto is "engaging audience makes 80 percent of problems solved".

Figure 10 Average User Persona

Business Analyst Jessy

Age	35
Job Title	Business Analyst
IT Experience	Advanced
User Group	Power Users

BACKGROUND

Jessy is a business strategist and decision adviser dealing with data analysis in their company. Jessy is majored in business school. She's not a programmer, but she's good at Excel and playing with data analyzing and reporting tools. She is quite familiar with different data tools and has a good relation with her company's IT department. She has the responsibility to produce data reports and analyses for managers and help them in making strategic decisions.

MOTIVATION

Jessy spends a lot of time tracking and playing with company's data. For her it is important to answer why some financial or business related phenomena happens to be able to help managers in their tasks. She usually gets raw data from IT department put them into excel data sheets and spend lots of time playing with data to find general trends and tries to understand relation between that data.

FRUSTRATIONS

Usually excel is a straightforward tool to fulfill some of her needs but at the same time it does not give her enough support, so she needs to use different tools but usually they are not that much compatible with each other which cause some difficulties for her. After using different tools she has found that some data tools are so simple that does not allow her to mine and browse data based on her needs and some tools are too much complicated that takes her ages to get to a simple result out of data. She needs complicated features but with simple workflows.

IDEAL EXPERIENCE

For her the quality and accuracy of data is really important. She is found of fast and detailed data interaction and visualization techniques. Mostly she needs some rich but simple tooling sets to help her first in finding the right data then realizing and analyzing the important trends and relations between data and finally to help her in presenting her data analyses and results with some understandable presentations. For her learning curve is not that important, she can spend time to get master in a tool in case the tool support her with some advanced feature because she wants to use it in day to day life and being fast with her tools is important for her.

Figure 11 Advanced User Persona

3.3 Requirements of BI/SSBI Tools

BI and SSBI (tools) has many different kinds of functional, usability and technical requirements, in this section a list of high level requirements for BI and SSBI tools is mentioned.

3.3.1 Gartner BI Software Capacities

Gartner paper has defined 17 features or software deliverable capacities as requirement for BI software platforms. This 17 capacities are classified in three groups as *information delivery*, *analysis* and *integration*. In the below these 17 high level requirements are discussed. [44]

Group1. Information Delivery

3.3.1.1 Reporting

The most obvious high level requirement and capacity is report production. Still designing, browsing and printing reports is one of the most basic users' needs from a BI tool. BI tool should empower its users to be able to create interactive, formatted, and also print-ready reports. [44]

3.3.1.2 Dashboards

The concept of dashboard is quite new but it is ubiquitous like data itself and it can be seen in many applications whenever it is needed to monitor changes in data. There are many definitions for dashboard, *Peter McFadden* defines it *"Dashboard reporting is the process of designing an easy to read, often single page, real-time user interface, showing a graphical presentation of the current status (snapshot) and historical trends of an organization's Key Performance Indicators (KPIs) to enable instantaneous and informed decisions to be made at a glance* [47]*."*

When users want to monitor changes on some data they like to have a home page or view in which they can have a holistic view of total trends (trends that they are interested in) of their data, they can (with one glance) see changes and start drilling down to important parts. *Gartner* itself defines this ability in details as *"a style of reporting that graphically depicts performances measures. Includes the ability to publish multi-object, linked reports and parameters with intuitive and interactive displays; dashboards often employ visualization components such as gauges, sliders, checkboxes and maps, and are often used to show the actual value of the measure compared to a goal or target value. Dashboards can represent operational or strategic information* [43, p. 2]*."*

Besides being useful to have something that convey information fast; the other important feature of dashboards is that they usually are built with gadgets having a sleek graphical design with engaging visualization and animations attracting users who like to have fun with data while interacting with it (again the brilliant quote of *Norman "Attractive things work better* [31]").

3.3.1.3 Ad hoc report/query

As it was explained before for a BI tool it is important that users be able to play with data and design their report without having a deep knowledge of database structure. Ad hoc reporting allows users to browse data storages and design reports without needing to deal (directly) with the data models. Gartner defines this capacity as enabling *"users to ask their own questions of the data, without relying on IT to create a report. In particular, the tools must have a reusable semantic layer to enable users to navigate available data sources, predefined metrics, and hierarchies and so on* [43, p. 2]*."*

3.3.1.4 Microsoft Office integration

Microsoft Office (particularly Excel) historically work as a reporting and data analytic client and also it is used for documentation and presentation (Power Point). As a result, it is an important feature for a BI tool to be able to work gracefully with Microsoft Office tools.

3.3.1.5 Mobile BI

Today mobile phones equipped with many sensors are everywhere, from one side they are an important source of data, and on the other side they could be used to browse data reports and analyses. Mobile phones and tablets applications could be a first class citizen to handle daily tasks so a full featured BI tool should give the ability of using mobile phones and tablets to get business insight in data. Formally mobile BI could be defined as *"the capability that enables the mobile workforce to gain business insights through information analysis using applications optimized for mobile devices* [48, p. 2]*."*

Group2. Analysis

3.3.1.6 Interactive visualization

Visualization and data interaction in a way to both amuse users and make it easier for them to find regulations and trends from data (which is an important part of data analysis) is one of the most important requirement of a BI tool. BI tools try to allow users to explore data by manipulating chart images with different color, various level of brightness, different sizes, shapes and animation representing different aspects of the data. There are quite many visualization techniques have been being used in BI tools (and also there are many researches to find more innovative ways to help users interact with and visualize data) including pie, bar, line, bubble, column charts, heat and tree maps, geographic maps, scatter plots and etc. [44]

3.3.1.7 Search-based data discovery

The quite new feature for modern BI tool is ability to search all data (structured and non-structured) with a search based interaction, for example users should be able to ask against datasets about facts by text and BI tool should be able to answer them by some visualizations of result data. One of the brilliant and innovate implementation of this feature is *Q&A* (question and answer) in *Microsoft PowerBI* tool chain. Based on its website *"Power BI Q&A presents your answers in the form of visualizations. As you type a question, Power BI Q&A picks the best visual to display your answer; and the visual changes*

dynamically as you modify the question. And you can further customize the visual results as well, using features like filtering and fields lists [49]*."* In Figure 12 one screen shot of *Microsoft Power BI Q&A* is shown, in the above textbox user asked question about the data, system offered users some related questions (to help the user to refine his question) and also the result of first question with a best matched visualization is drawn.

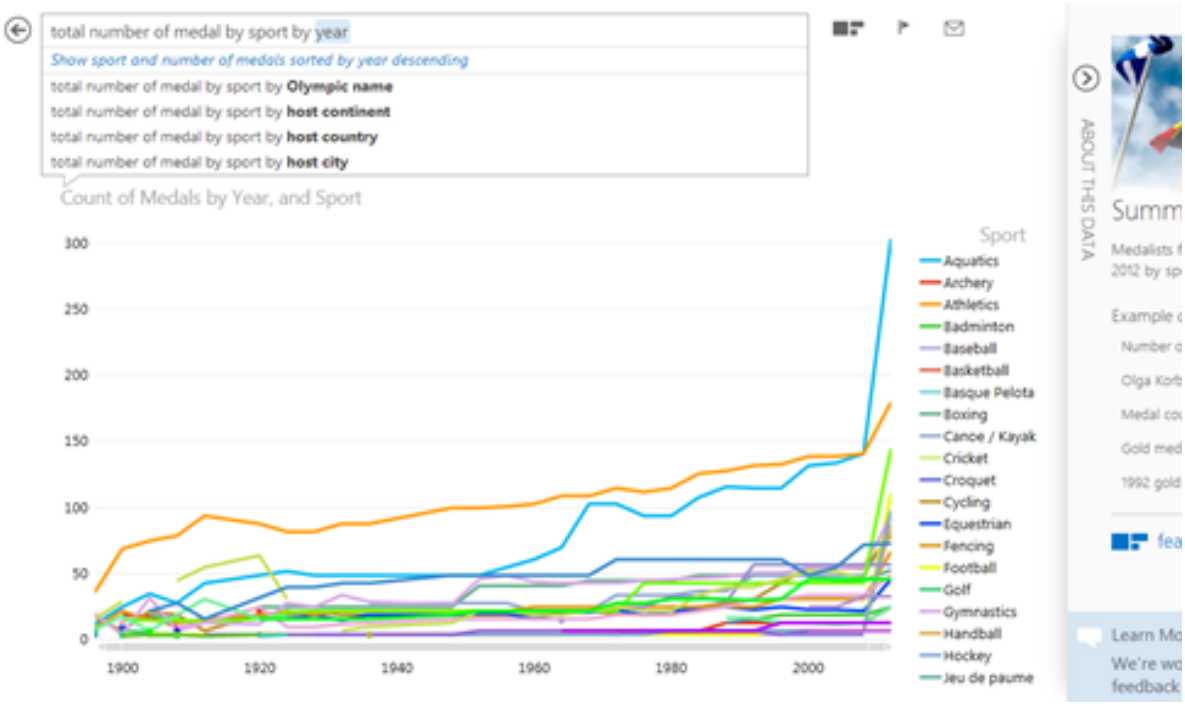

Figure 12 Microsoft Power BI Q&A

3.3.1.8 Geospatial and location intelligence

Usually lots of amount of data are connected to geo information and for whom wants to analyze that data it is really helpful to be able to see geo location related information in a visual intractable way. Gartner 2014 put more emphasis on this feature and locate it in a separate section requirement for a BI tool. *"More advanced capabilities support specialized geospatial algorithms (for example, for distance and route calculations), as well as layering of geospatial data on to custom base maps, markers, heat maps and temporal maps, supporting clustering, geofencing and 3D visualizations* [43, p. 2]*."*

3.3.1.9 Embedded advanced analytics

This ability is more about embedding advanced algorithm for predicting the future of a business inside of a BI tool to empower users get benefits of current and old analyses to predicate future and do the right things now. [35]

3.3.1.10 Online analytical processing (OLAP)

Supporting users with an OLAP tool and the ability to connect and retrieve data from OLAP servers is something inseparable from a BI tool. [35]

Group3. Integration

3.3.1.11 BI infrastructure and administration

BI tools are composed of different components, so all should have the same use and feel meaning integrated and also there should be a central administration tool to be able to perform administration level tasks like security, task scheduling and etc. [35]

3.3.1.12 Metadata management

There should be some capacities to give ability to administrators to manage meta data objects all around BI systems. *BI* systems *"... should provide a robust and centralized way for administrators to search, capture, store, reuse and publish metadata objects, such as dimensions, hierarchies, measures, performance metrics/key performance indicators (KPIs), and report layout objects, parameters and so on. Administrators should have the ability to promote a business-user-defined data mashup and metadata to the systems-of-record metadata* [43, pp. 2,3]*."*

3.3.1.13 Business user data mashup and modeling

BI tools should give abilities to users to be able to do some kind of simple data modeling, combing different data sources, refining and shaping data sets based on their needs (creating user-defined measures, sets, groups and hierarchies) with an easy to use drag and drop and code less techniques. [44]

3.3.1.14 Development tools

Beside trivial tasks (majority of analyses) which are done with easy to use interfaces, BI platform should support a full featured development environment for more advanced tasks, some facilities to customize tool and advance features by programming the tool itself. [44]

3.3.1.15 Embeddable analytics

There should be some capacities to enable users to put Business Intelligence (a software developer kit with APIs for creating and modifying analytic contents, visualizations and

applications) directly inside company's information systems like customer relationship management (CRM), enterprise resource planning (ERP) and etc. This feature allows integration between BI and internal information system of companies. [44]

"Embedding business intelligence (BI) and analytics capabilities directly into a business application helps users to benefit from actionable insights and information in the context of their usual workflow [50]*."*

3.3.1.16 Collaboration

It is a quite obvious requirement, users should be able to share, discuss, annotate and reuse data, data models, analytics and visualizations.

3.3.1.17 Support for big data sources

The ability to connect to big data sources and NoSQL storages.

3.3.2 SSBI Requirement based on Forrester

Beside the above list of BI software capacities, *"The Forrester Wave™: Self-Service Business Intelligence Platforms, Q2 2012"* published another list of requirements for a BI tool to be considered as a SSBI. Some mentioned features are the same as above list but with more emphasis on the usability and some are not mentioned in *Gartner* list, in the below some of these requirements are discussed.

3.3.2.1 AutoModeling

Considering the same capacities mentioned under ***Business user data mashup and modeling*** *and* ***Ad hoc report/query*** headline in the *Gartner* list, generally it is pointing to help users to model data and hierarchies from star schemas and connect datasets to each other implicitly by some kind of heuristics like connecting tables which have columns with the same names, recognizing hierarchies and automatically geocode attributes (such as ZIP codes, countries, regions, cities, and states), extracting measures and fact based on the a column's data type and etc. [40]

3.3.2.2 Calculated measures

(Again could be seen under ***Business user data mashup and modeling*** *and* ***Ad hoc report/query*** of *Gartner* requirement list) For IT developers it is not possible to predict all the possible measures and calculations to predesign them inside their OLAP system, as a result to allow business developers use SSBI tool without needing to ask help from IT

developers to add new calculations; there should be some easy ways to add calculations to data models by end users. Some trivial calculations like adding a C column which is calculated based on the value of A and B column something like C = A+-/*B or defining cumulative calculations like C column in each record calculated as the total of the value of A column of the previous records + A's value in this current row and etc. [40]

3.3.2.3 Collaboration

The same as it was mentioned in Gartner list by more emphasis on sharing results in the other communities like Facebook or other company's internal systems like SharePoint. [40]

3.3.2.4 Data virtualization and drill anywhere

Users of BI and specially SSBI should have the ability to go to details of data and get as much information as possible. It is one of the subtle but the most important feature for a SSBI to provide fast, agile and independent experience of browsing data for its users. In this resource this feature is defined as *"In today's world of big data, no one can afford to put anywhere close to 100% of their enterprise data and information content into a structured database. Therefore, capabilities to virtually link multiple data sources and drill anywhere within these sources becomes important, unless one can afford to wait for days, weeks, or months for these new data sources to be loaded into an enterprise data warehouses (EDW)* [40, p. 3]*."*

3.3.2.5 Prompt for column

This feature is about using report templates and defining columns as parameters then report designers easily will be able to change the columns' names based on their needs and design what they wanted as fast as possible. [40]

3.3.2.6 Search-like GUI

This requirement is the same as Gartner list and more emphasis on usability. Based on Forrester this kind of user interface provides some benefits as *"First, it requires little to no training; hardly any knowledge workers in the modern world can't use a web-based search tool. This is a benefit, as every hour of BI training for a salesperson is an hour away from customers. Second, a point-and-click GUI assumes that you know exactly what you are looking for, while very often you don't, and a search-like GUI is especially handy in you-don't-know-what-you-don't-know scenarios Last but not least, search-like*

GUI enables faceted navigation, which is often the best way to drill up/down/across hierarchies that are ragged and unbalanced. [40, p. 4]*"*

3.3.2.7 Application sandboxes

Again because SSBI tries to support end users without any (or less) needs of referring to IT partners it is important to have sandbox like version of BI on their personal system. It allows users to protect themselves from BI versioning hassles. [40]

3.3.2.8 Migration to production

It should not be a hard job for ordinary users to deploy and publish their reports and analytics into shared environment or production state. There should be something like one click publishing to allow business users to deploy their result into production environment. [40]

3.3.3 Requirements Summary

One common and important point in all these requirements is the emphasis on the users. As it was discussed in the previous chapters, considering ordinary users and providing a good experience for them in dealing and analyzing data is the spirit of a BI/SSBI tool.

The functional and nonfunctional requirements mentioned in the above lists are a sample of standard BI/SSBI requirements, so a full-fledged BI/SSBI should fulfill the most of it though implementing this long and advanced list of requirements needs considerable resources. For this thesis work few most important of the above requirements are selected and implemented this short list of requirements are discussed in the next chapter.

4 DESIGN OVERVIEW OF THE PRODUCT

This product is based on real customer's needs of ***Wapice Oy***. Customers have had difficulty to adopt to current BI applications in the market made using current BI/SSBI applications not as efficient as they expected, so it was decided to develop a small scale and simple version of a reporting tool (SSBI tool) to fulfil customers' needs. It was planned to have a prototype version of the application in the first phase of development process to reduce the risk of heavy investment and start a big application from the ground without analyzing it in depth.

As it was described in the above chapters, BI/SSBI tools are complicated applications and they should fulfil a long list of functional and usability requirements. So, developing a full featured BI tool takes several years and needs a quite big and mature software and UX team. In this first version the goal was preparing a small version (more like a prototype application) to be able to evaluate technical and usability decisions for future more mature versions, it was tried to limit the scope of the application and focus on the most important requirements and features.

In this chapter first the focus user groups of the product is explained then the most important requirements of SSBI which is concerned in this first version of the application is mentioned after that most important design decisions about technical and usability solutions are discussed.

4.1 Focused User Groups

In the chapter 3 a holistic view of possible users groups of a SSBI/BI tool was discussed, from the above list the focus for this first version of the product was on two groups:

- The first group includes people who just want to view (browse) prepared interactive reports, this group usually do not need (or do not have enough time or technical skills) to design reports, they retrieve reports from a report storage from the report server and usually try to understand trends of data based on the current designed reports. Formally this group are part of casual and executive users groups which mentioned in the previous chapter.
- The second user group targets people who use the application to design interactive reports, who are savvy IT people which do not necessary have coding knowledge and experience, they might have just responsibility to prepare reports for the first group or they might themselves need to analyze data in a fast and easy way. This group is a mixture of IT developers, power users and average users from user groups mentioned in the previous chapter.

In the result application there will be a report viewer application which targets the first user group and a report designer application which will target the second group. It is tried to have consistency in the both viewer and designer applications in all levels (in both

UI and code level). Figure 13 shows the main components of the product and their relation to user groups.

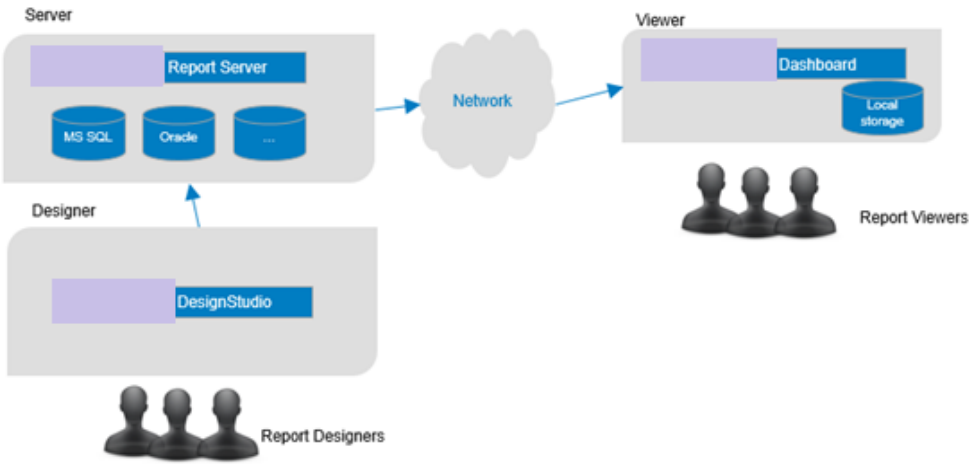

Figure 13 User groups and components

Figure 14 is a high level use case diagram and is showing the most important uses cases: designing, viewing, analyzing and publishing reports, browsing report folders and users authentication (other use cases are not shown in this diagram).

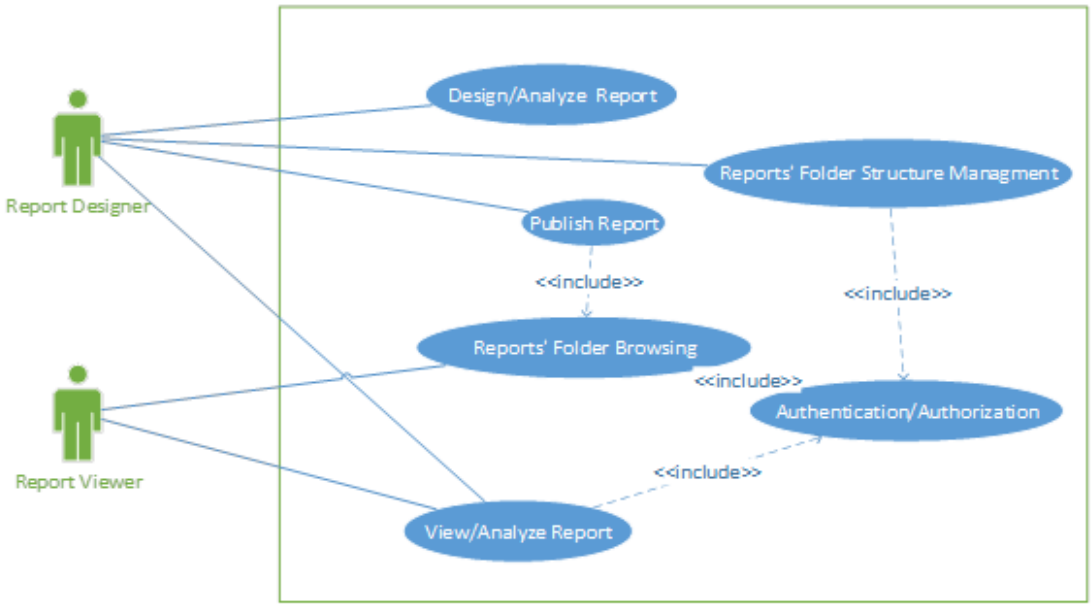

Figure 14 Main use case diagram of the product

4.2 Selected Requirements

In chapter 3 a list of formal requirements for BI and SSBI tools were mentioned. In this first version of the product it was decided to implement the most important usability and

functional requirements (with cheapest technical cost). It was important to produce something as a working prototype that could present the main idea of the application. The below list is showing the selected requirement for the first version of the product.

- **Interactive Ad hoc reporting:** The main high level requirement to cover (targeting report designers' user group) is providing the ability to have interactive ad hoc reporting. Users should be able to produce their reports just by drag and drop. Users should be able to add visualizations based on their needs to their reports, customize them and connect those visualization to data by some mouse clicks.
- **Interactive report browsing experience (dynamic filtering, sub reports, data grouping and drill everywhere):** The other important requirement which the application should fulfil is providing an interactive report viewing experience for its users who browse prepared reports. Those static *pdf* based print ready reports are good just for documents but not for understanding data. Users need reports which allow them to drill into and filter data while viewing. When report designers design reports they should design enough interactions to allow viewers freely go to details of data whenever it is needed and filter report data based on their needs. Report designer part of the application should allow designers to design interactions (for end users) inside reports and ReportViewer application should allow users to play with data and visualizations in designed reports.
- **Export reports in presentation format:** It is quite related to *"Microsoft Office integration"* from Granter list, customers like to see that their reporting tool is able to export reports as presentation (like Microsoft PowerPoint format). Because this feature is not expensive to implement and at the same time has a noticeable impact for the marketing aspects of the product, this feature is implemented in this version.
- **Calculated measures:** For report designers this requirement is an important one. This feature allows report designers to design their own data model just by drag and drop without knowledge of programming. It allows them to shape data in a way that they need. Reports designers are not supposed to change the original data model (by Sql or ETL) but should have the ability to customize retrieved data and build a custom version of data model.
- **Easy report publishing:** This requirement covers Migration to production requirement from Forrester research paper. There should be an easy way for report designers to publish their designed report.
- **Security and Profile:** There should be some report level security and profile management to allow report authentications and home page customizations for users (decided but not implemented at end of phase one).
- **Support different data storages:** The application should be able to connect and use different data storages like *MS SqlServer*, *MySql*, *Excel*, *Oracle* and etc.
- **Offline Storage:** Report viewer should work in offline mode. Users are able to save reports and load them without needing to be connected to data servers.

4.3 Design Decisions

In developing any business applications, there are some critical design decisions could have big effects on the result of software production process (even can result in success or failure of development process), decisions like choosing development technology, tools and programming languages, user interface interaction techniques and etc. are part of those application level decisions. In this section the most important design decisions of the product under development are discussed.

4.3.1 HTML/JavaScript as Development Platform

Final version of the product should support different platforms. It should support tablet and mobile devices as well as desktop and web interfaces. So for developing this solution there was to different choices: either development team should design and develop native applications for different environment (different mobile applications and probably a web version) or the second solution is developing one HTML based version that could run in any environment. In the big road map for the product it was decided that it will be a HTML based environment to reduce the cost of development and to be able to provide the same and integrated experience for users in different environments.

As a result to be able to use the code of this first version in the next versions it was decided to develop this version as close as possible to the final version meaning using HTML and JavaScript (HTML and JavaScript hosted in a desktop window application) in this version as much as possible. It was decided to have a hybrid HTML and window native solution for the first version of the application. In the front end of the product, there are two separate applications one report designer and the other report viewer. These two applications are window desktop applications hosting report pages so some parts of the logic of the application is happening in the native environment and some part of the application is happening in HTML. In the next section more details of this technical design are discussed but for this section it is important to mention that the main part of the project is done with JavaScript and HTML.

From end users' point of view, designers have a window application and design their reports (in HTML format) and publish them to the server after that, report viewers retrieve those reports (web pages) and view them on the viewer application which is again a window application hosting report HTML pages.

4.3.2 Development Tools

Based on the experience of development team *.net* was selected as the main programming environment both for desktop applications and web server side. Microsoft development environment meaning Visual Studio as development tools and C# as programming langue usually are one of the best choices for developing business application.

4.3.3 Visual Interaction

In the report designer application, there are two designer panels in one users are able to design the visual part of their reports, how a report is structured, how different items look, how they are located beside each other and etc. In the other designer panel users are able to design the data follow in the report. In this section the first panel is described and the next section will explain more about data designer panel.

The report designer application has a tool box and a design space. Users are able to drag design items from tool box and drop them on the designer panel and design their reports. There are 2 different types of design components:

1. Components which allow users to build the structure of their report. They can use this group of components to divide a report and make a group of components. These grouping components are resizable and users are able to change the size of each group. Group box, TabControl, Panel, Table and etc. could be some samples of these kinds of controls.
2. Components which have the responsibility to visualize data from datasets. All different kinds of charts (Pie, Bar, Column, Line…), Geo Map, DataGrid, all kinds of ListBoxes (radio list, checkbox list), ComboBoxes, even Labels, Buttons and etc. These controls will be located inside the container controls (the above list) and users assign them some data to visualize, this data could come from different data sources (technically data binding sources) or can be assigned to controls statically like a label box which shows a static text, also these values could be a result of a calculation on some data like a label box which shows the number of records. The most important feature of these design items is that they are interactive and designers can assign them some actions then in the viewing time, viewers are able to fire those actions and do what they want (for example a report designers might give a filter action to a button then in the run time, report viewers can select some value then click that button and a related chart will be filtered as the result of this action).

Both of the containers and controls has their own (specific or generic) properties which users are able to set. Usually these components have two different parts; visual part and data part. So users are able to set properties related to visual part of design items in this space (UI designer panel) and use the data designer space to set properties related to data functions of these components. In Figure 15 you can see this UI designer panel and visual components. A tab control shaped the report page, then a data grid is showing some data and at the bottom of the report one pie chart and one line chart are placed beside each other in one group box.

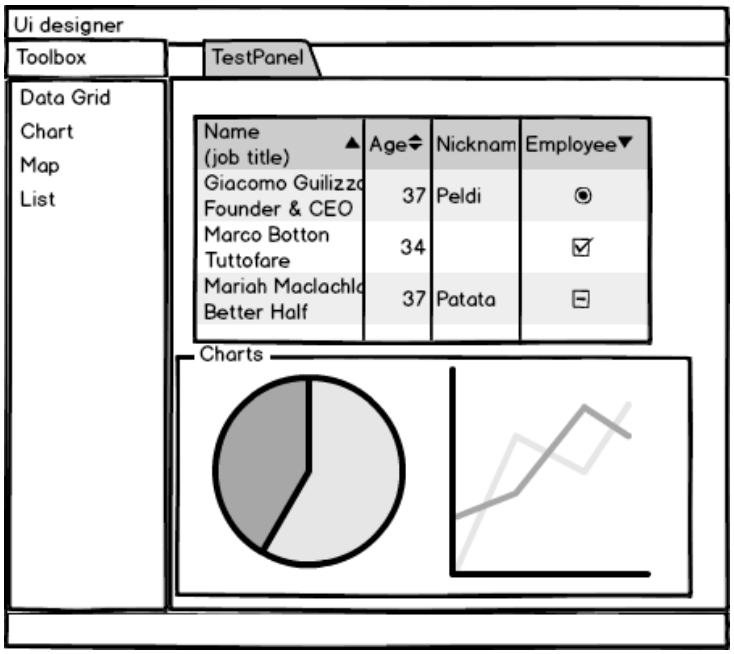

Figure 15 UI designer's view

4.3.4 Data Interaction Techniques

One critical part was designing an intuitive for users to design their data model and assign data to visual components in the data designer part of the application. As it was explained in the above section, UI design items has a counterpart data design item and there is a panel in which people can design the data part of a UI object also the data flow in the whole report. In this panel people can design three different stuff, first the data flow of the report: how data goes inside controls, the second define action for controls: what should happen when an action get fired and third they can customize data and design their own data model.

This panel is quite like a graph and each data design object has a visual representation as a node in this graph. Users can connect these nodes to each other which this connections show how data should flow from one node to another. There will be different kinds of nodes like data sources (which will be set to some data storages like Excel, MySQL …) and then users can connect data sources to other nodes like a data grid, so that data grid will retrieve data from data sources and will show it. Users add data sources, connect them to data storages and use them as a source of data for other components.

Some data nodes like datagrids or charts just show the data they receive from input but some other nodes like data shapers (technically BindingSource nodes) can shape and change input data and produce new data as output. By this users are able to wire up their report by data and at the same time they are able to design their data model interactively. For example in Figure 16 a user added a search text box in the UI designer, then in the data designer he has a filter node, this node gets some data from input and sends the result to a data gird. At the run time, when a report viewer will add some text inside this filter box and press enter, this item will retrieve data from the previous node then it filters

that data based on the input text and then will send the result it to the data grid. Figure 16 shows a design for this concept.

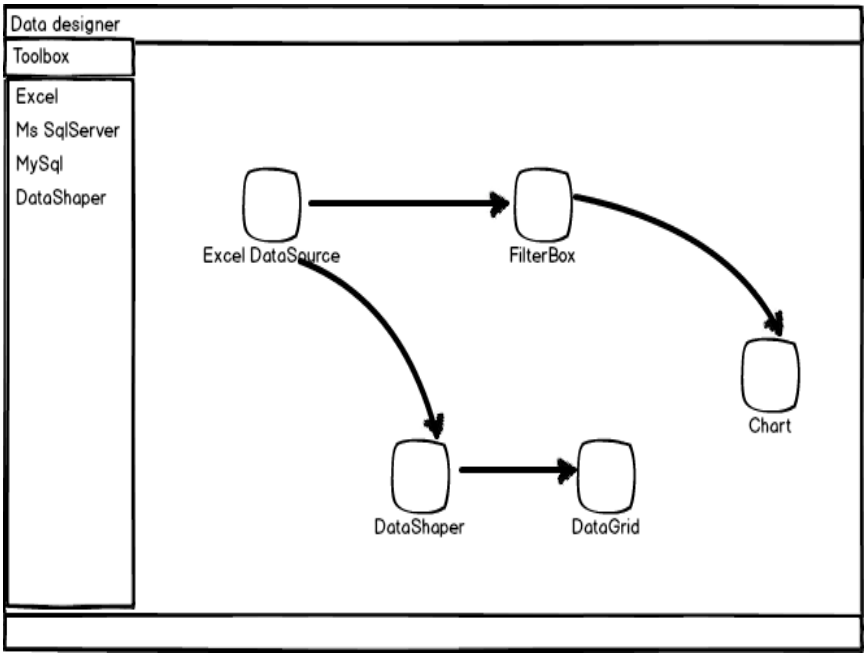

Figure 16 Designing data flow (data interaction)

One of the most important nodes here are data shapers (or technically named Binding Sources) these nodes allow users to shape incoming data and produce new data (modeling, refining data). These data shapers allow users to do whatever a database programmer can do in a ***select*** statement. This allows users to produce new calculated or cumulative columns, allowing them to group coming data and produce aggregated columns and generally allowing them to shape their data. In the below sketched data designer panel a user dragged the column A and B (from incoming data to a data shaper) and drop them into the editor panel, then he dragged a Calculation node and drew two edges from A and B to that node and took another edge from Calculation node and connected it to a calculated column named C, by this he added a new column to exiting incoming data and made a new data in the output. You can see a design of this concept in Figure 17.

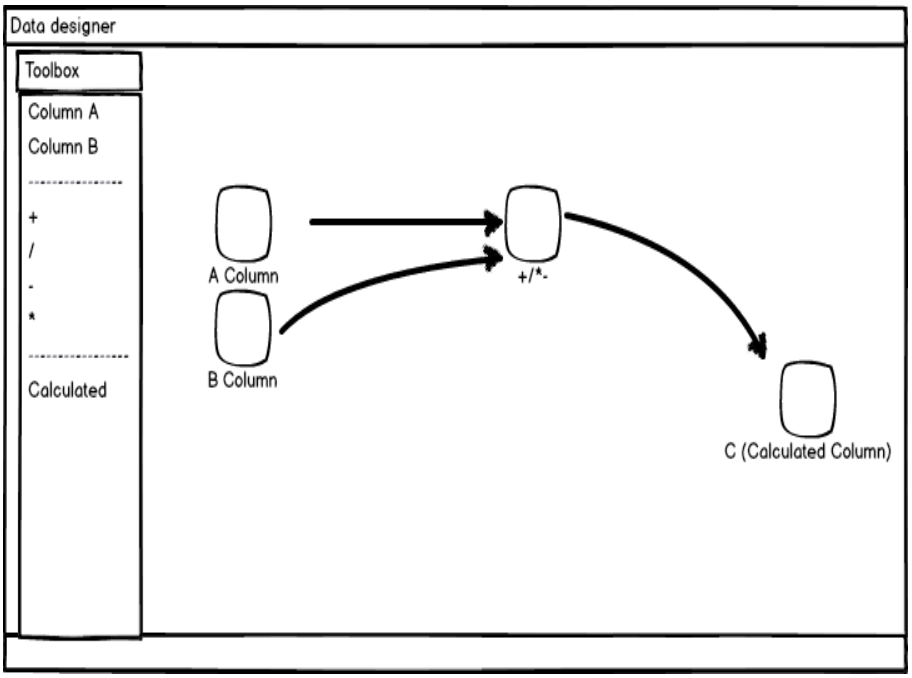

Figure 17 Customizing data model in a data shaper (data interaction)

Finally Figure 18 is showing the usage of data designer in designing a drill action. Designer of the report drew a click edge from a chart node to a sub report node, this tells to the application when a user clicked inside that chart, it should show subrpoert1 as the result of the click action, by this report designers can design drill actions in their reports (report designers should design UI and data of sub reports in the same way as the main report, also they can set which data should pass to a sub report while connecting it to a BindingSource node).

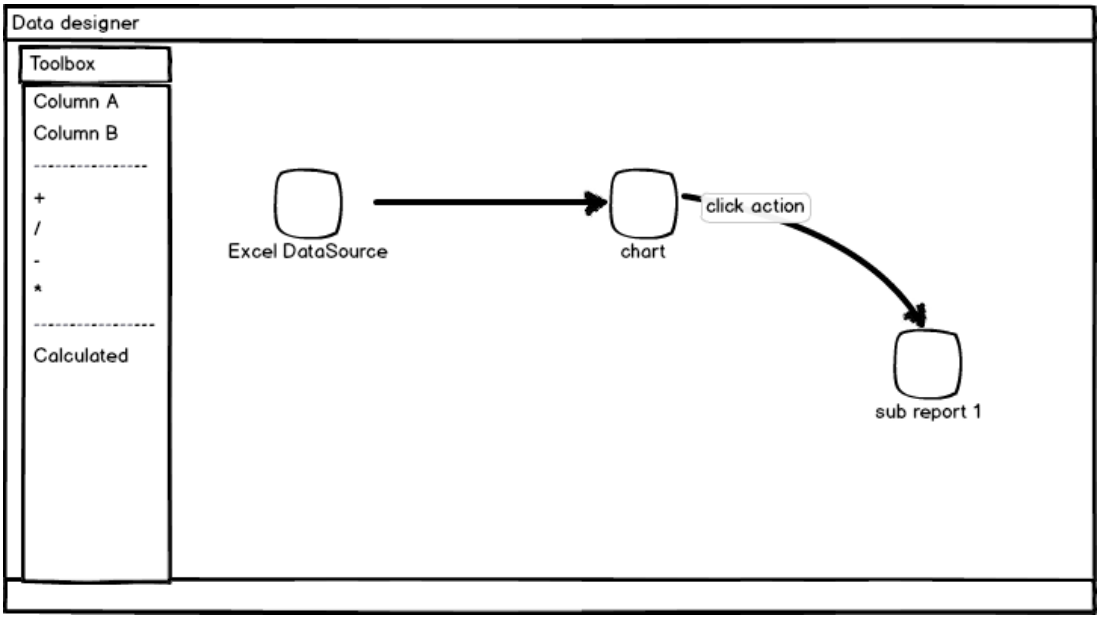

Figure 18 Designing actions (data interaction)

In the 6th chapter these design decisions are evaluated deeply to be able to see technical and usability problems of these designs.

5 IMPLEMENTATION OF THE PRODUCT

This product is implemented as a hybrid web/windows application. In the web part HTML and JavaScript (also some JavaScript libraries like jQuery, jQueryUI, HighChart and etc.) are used. In windows part C# as programming language, WPF (Windows Presentation Foundation) as user interface technology, RESTFul WebAPI in server side and VisualStudio as a robust LOB application development IDE was used (.net technologies like *Entity Framework* and *WCF* is used as well). In this chapter some technical details of project design and implementation are discussed and some snap shots of the result product is shown.

5.1 High-level Architecture of the Application

This product is a reporting tool which provides the ability to design (and analyze) then view (and analyze) dynamic ad hoc reports for different user groups. This tool chain is composed of three component, **ReportDesigner** and **ReportViewer** in the front end and **ReportWebServer** in the backend. After designing dynamic reports with **ReportDesigner** application, published reports will be stored on **ReportWebServer** and can be viewed by **ReportViewer** (also users are able to save reports on their local machine then they are able to load and run them in an offline mode).

Figure 19 is showing a high level view of the solution. In this view all three different sub parts of the application are shown (to show the concept of this application data sources are shown as well, although they are not part of the solution).

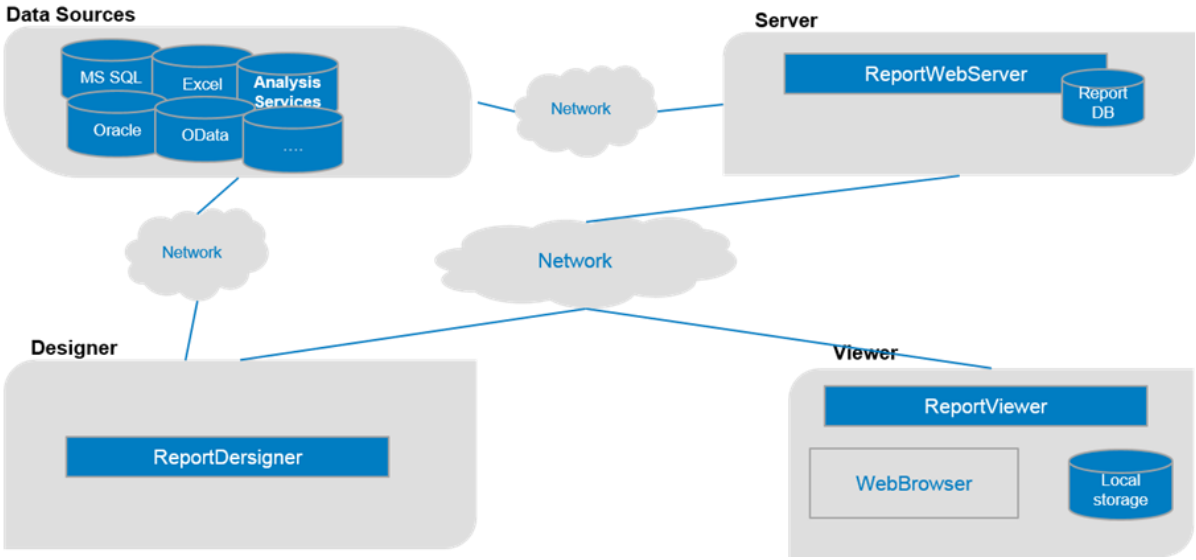

Figure 19 the product high level view

The solution is composed of four components plus some libraries and

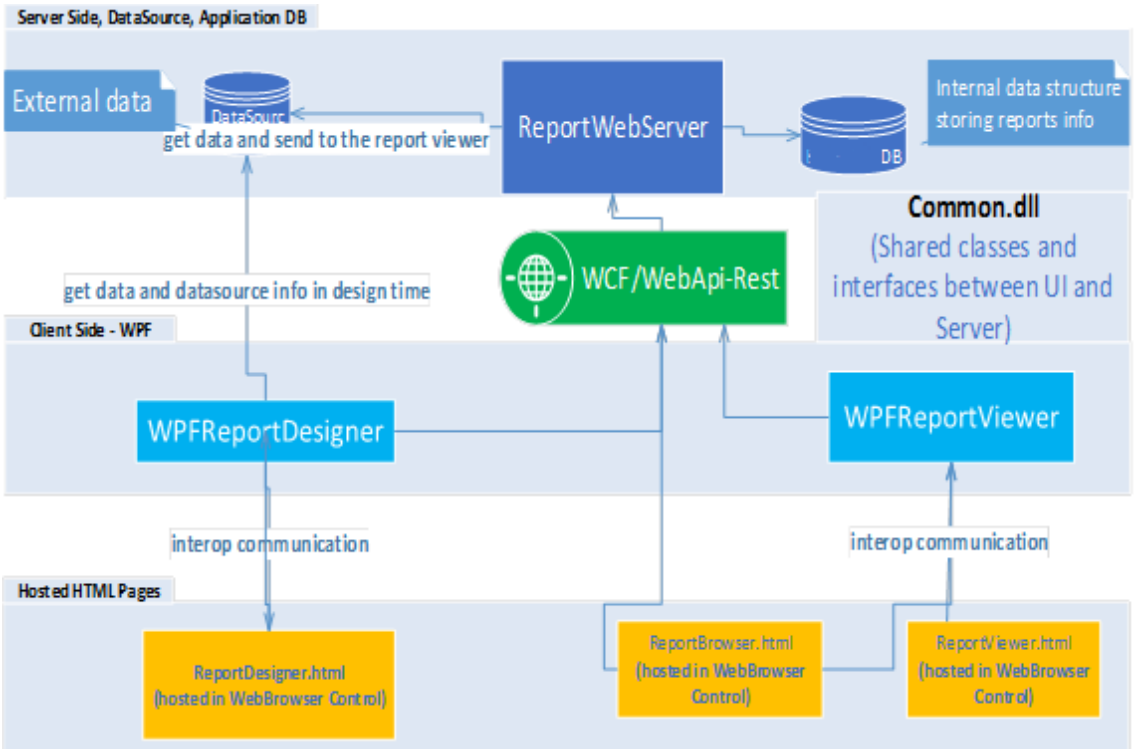

Figure 20 is one view of this architecture. The bottom layer has some HTML pages and JavaScript classes that are at the heart of the report designer and viewer, the second layer contains two WPF (Windows Presentation Foundation) projects which mainly host the web pages and provides facilities to design and see reports, the top most layer is a web application providing services for storing and retrieving reports, and finally a shared library used between the middle layer and server side application.

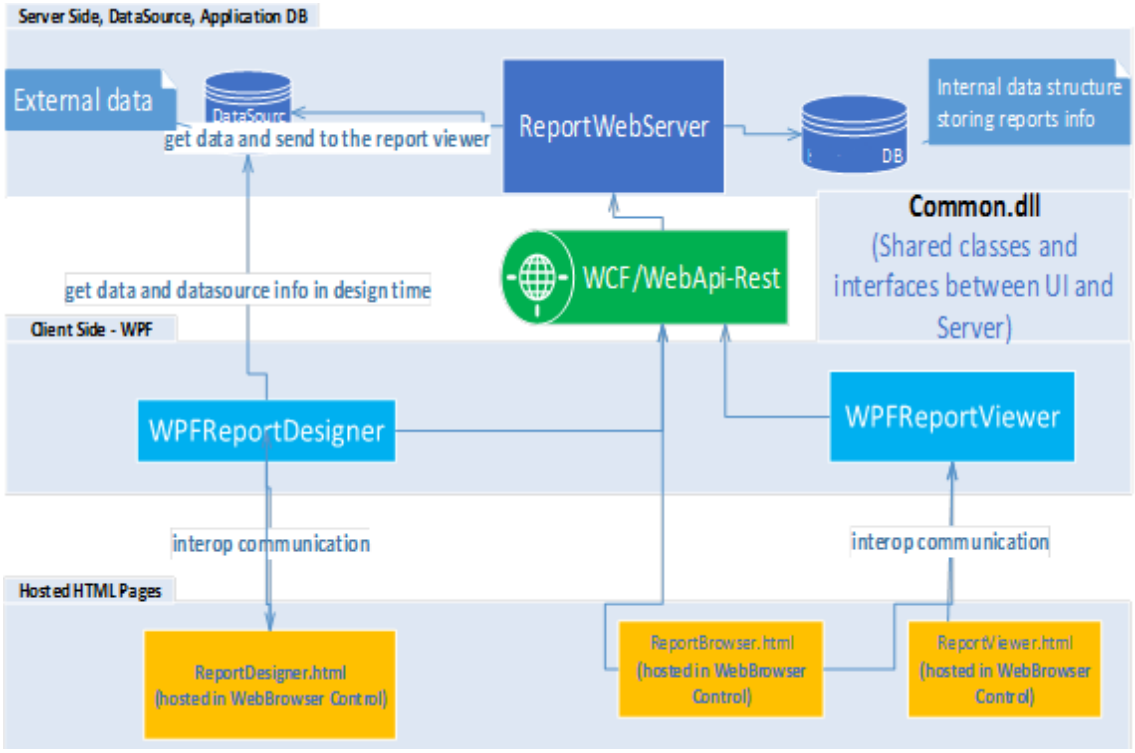

Figure 20 Architectural view

The next diagrams is showing the main work flow for two different user groups, the upper one is the use case of report designers (as a user) and the one at the bottom is the use case of report viewers.

Report designers will use *WPFReportViewer* application to design dynamic reports. They select their data sources and use that data to design their reports. After designing a report, users will publish their works to *ReportWebServer* so reports' meta-data and information of reports' data sources (connection strings and etc.) will be stored in the application database.

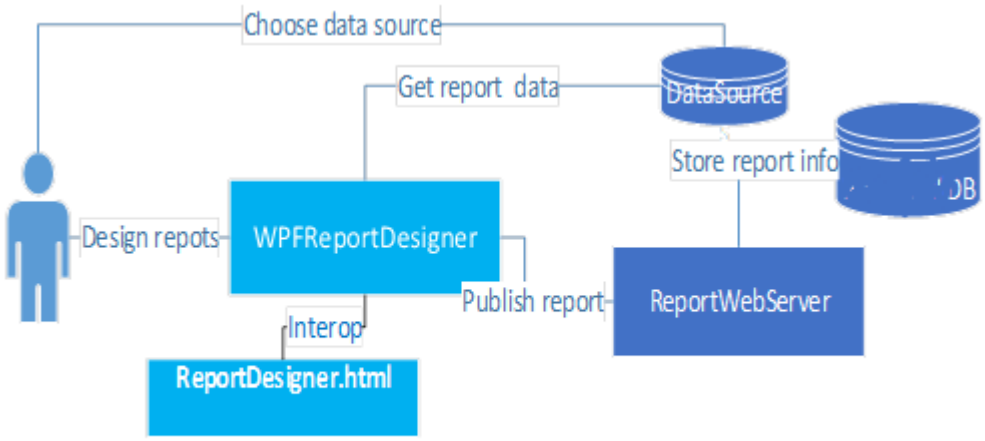

Figure 21 Report designer work flow

In the second scenario, users who want to see and use designed reports, first make a connection to *ReportWebServer* with *WPFReportViewer*. After that they will be able to see the list of reports on the server, so they can choose a report to view. Also in *WPFReportViewer* there is the ability to save reports and then load and work with them in the offline mode.

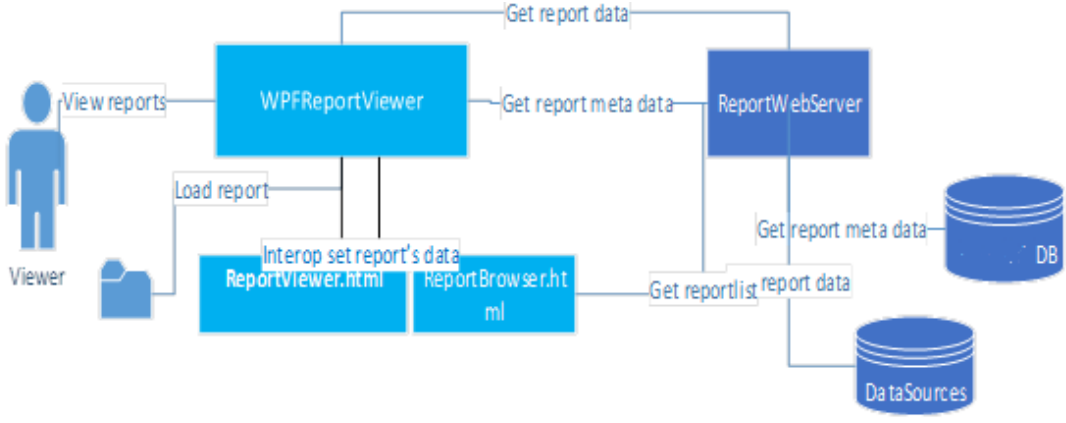

Figure 22 Report viewer work flow

5.2 Software Components

As it was explained the solution from technical view is composed of 5 different parts. *WebServer*, *ReportDesigner (WPF UI)*, *ReportViewer (WPF UI)*, *HTML/JS* components and finally some *shared libraries*. The .net solution for is composed of 8 modules with some third party libraries. Figure 23 shows a screen shot of .net modules of product solution.

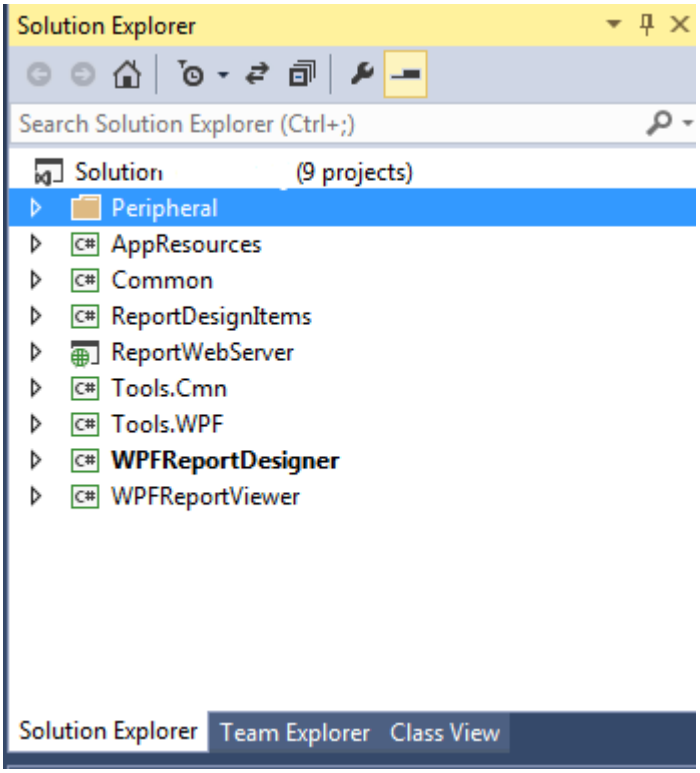

Figure 23 .net solution

Figure 24 is a dependency diagram for these application components.

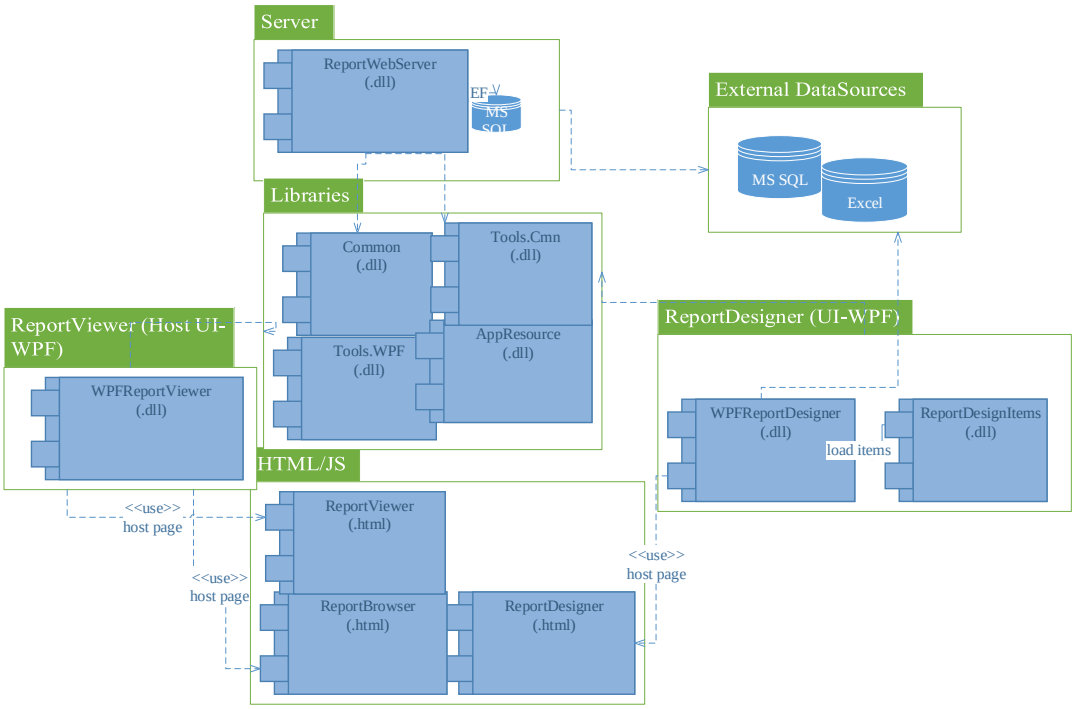

Figure 24 Component diagram

These 5 different modules are separately discussed in technical details in the next section.

5.2.1 Server

This module includes a web application *ReportWebServer* and a *MS SQL* database. The main responsibility of this module is providing functionalities for storing and retrieving report meta-data (reports' structure), and also retrieving reports' data from external data sources based on a client requests. The other responsibility which is **not completely implemented** in this version is user security, authentication, authorization and managing reports' permissions.

5.2.1.1 ReportWebServer Module

This module has two main namespaces:

- ***ReportWebServer.Data****:* containing POCO (Plain Old CLR Object) objects of each entity and also mapping classes.
- ***ReportWebServer.Code****:* containing services which are used by client applications (*WCF* and *WebAPI* services).

Also this project contains all *HTML* and *JavaScript* files physically (during development time). These HTML and JavaScript files will be copied in a folder named *web* beside both client applications (*WPFReportDesigner* and *WPFReportViewer*) to be hosted inside

them. More information about HTML pages and JavaScript classes are mentioned in a separate section.

5.2.1.2 Database

There are four tables to store folder structure of report repository and also reports' metadata. The *ReportContent* field in *Report* table contains reports' structure information in *json* format which is the json presentation of the report layout and serialized behavior of the designed report. Each report also could have different data sources which are hold in *DataSource* table in which the *DataSourceInfo* field is used to store connection and address of a data source.

5.2.2 ReportViewer

Project WPFReportViewer is located in this module. This is a WPF project hosting a WebBrowser control.

- Users can connect to the server and browse the list of reports from the server and choose a report from the list of reports.
- Users are able to take snap shots of different charts and save images or export snap shots to PowerPoint.
- Users can save reports in their local file system and load it later without a connection to the server (working in offline mode).

Figure 25 is showing the main window of report viewer application, here users can use main menu to connect to the report server and browse report folders.

Figure 25 Main window UI (report viewer)

Figure 26 shows how *ReportBrowser.html* web page is hosted in the report viewer application, through this view users are able to browse designed reports and select whatever report they want.

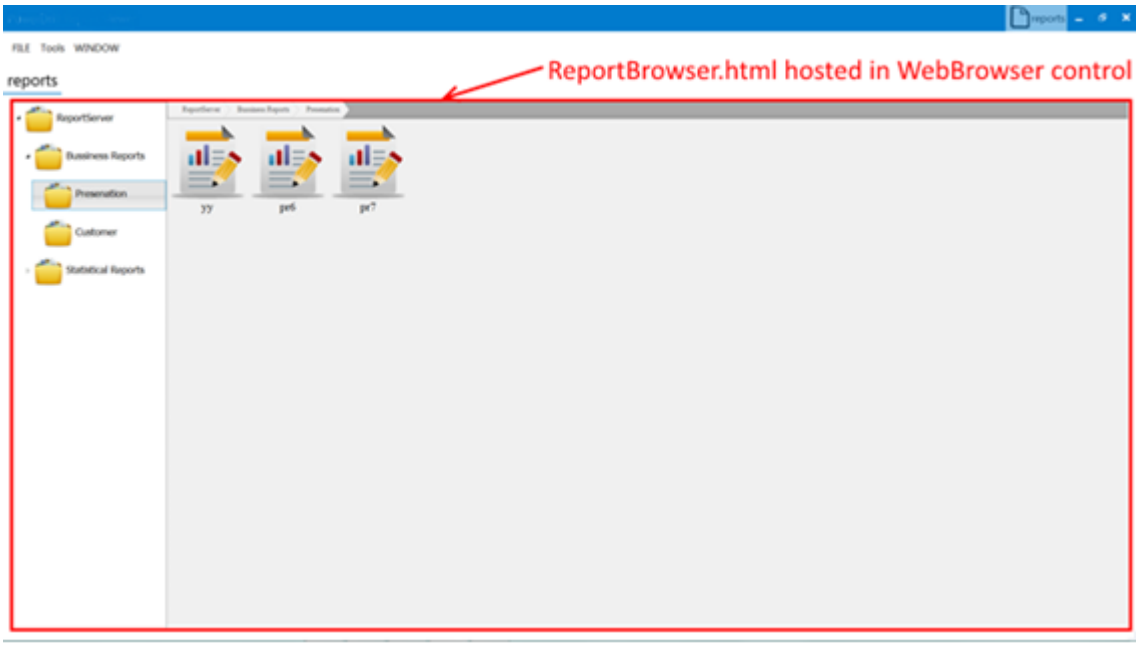

Figure 26 Report folder browser UI (report viewer)

In Figure 27 you can see a report is loaded in the report viewer, so users are able to play with the report, see visualized data in charts, filter data by select an item from a ComboBox, drill down to charts, save and load a local version of the report, take screen shots of charts and etc.

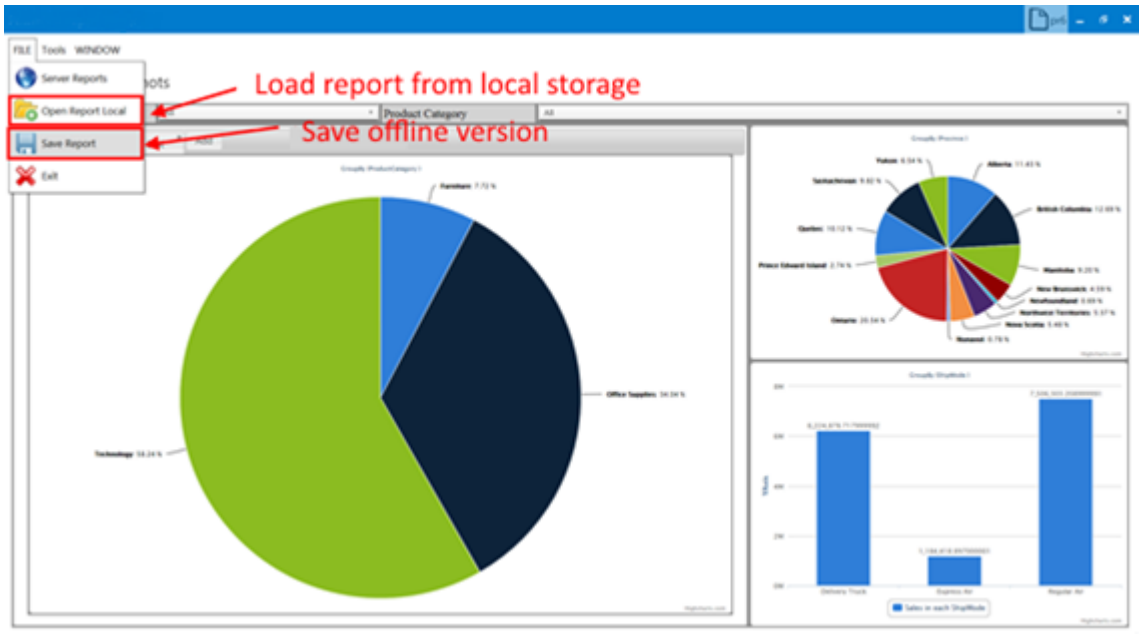

Figure 27 Loading report (report viewer)

And finally in the last screen shot the ability of managing screen shots and making a PowerPoint presentation from screen shots are shown.

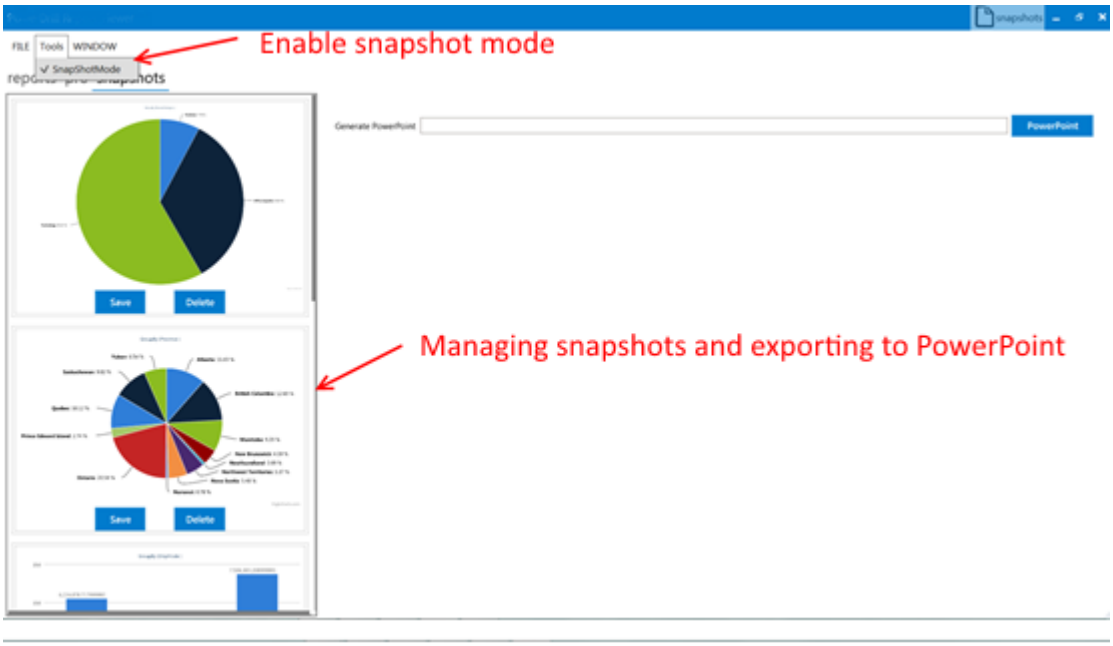

Figure 28 managing screen shots and making presentation (report viewer)

This application is done with WPF technology with MVVM (Model View View-Model) pattern. So each view has a corresponding view model which has the responsibility to manage that view. The main view has three panels: report viewer, report folder browser and snap shot manger. In Figure 29 the main class view of this application is shown (this class model is showing how different ViewModel classes are related to each other).

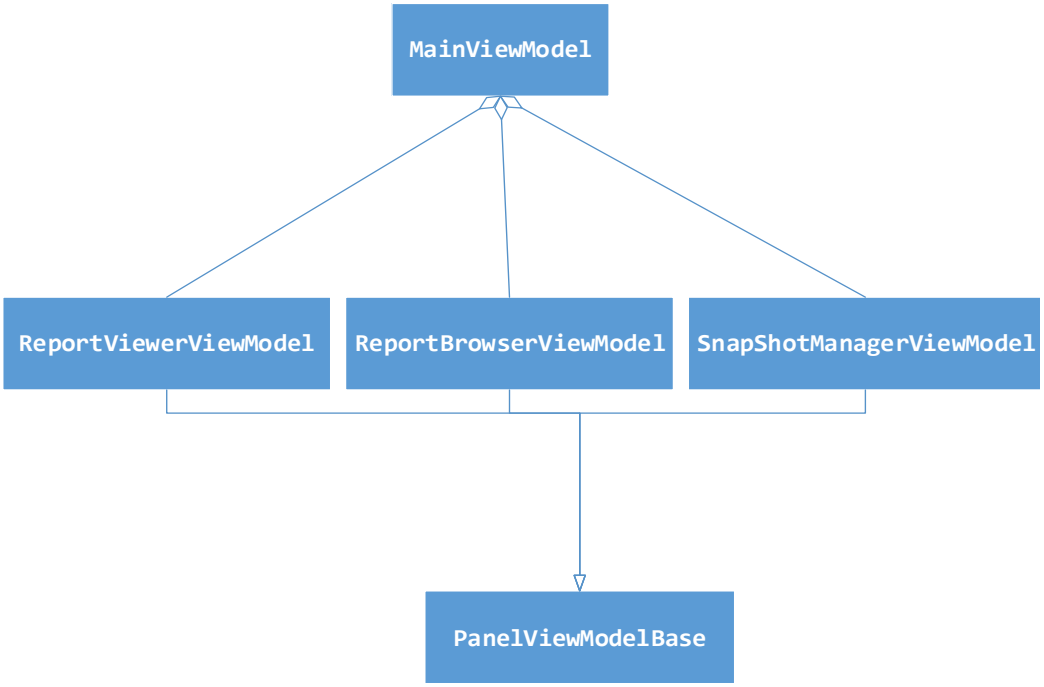

Figure 29 Main class model (report viewer)

5.2.3 ReportDesigner

In this module *WPFReportDesigner* and *ReportDesignItems* projects are located.

5.2.3.1 WPFReportDesigner

This module brings below functionalities to the application:

- Users are able to design their reports
- Users can see the result while they are designing their reports at the same time
- Users are able to publish their reports to the server
- Users can save/load a design version of reports in the client

In the main window there is a workspace view which have all the stuff needed to draw reports. There are two designer panels one for designing UI part of a report and a data designer for designing data flow (users can select each panel). On the left there is a toolbox and on the right side there is one report outline and one property panel. In the below screen shot you can see the UI designer panel.

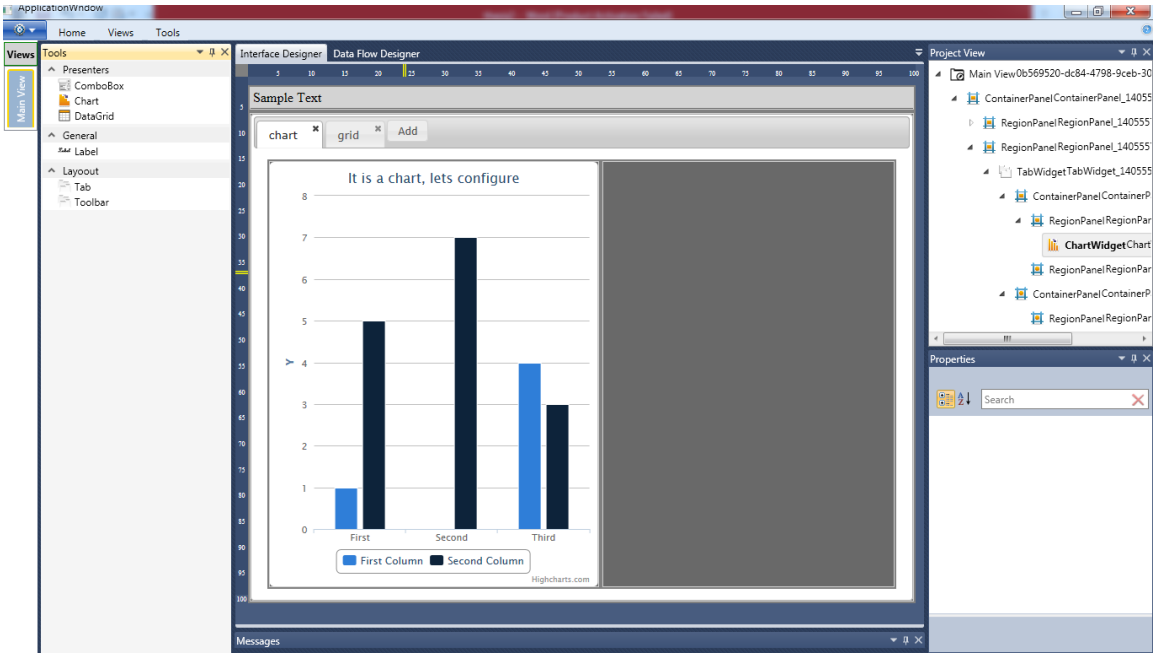

Figure 30 UI designer (report designer)

Figure 31 is showing the data flow designer panel.

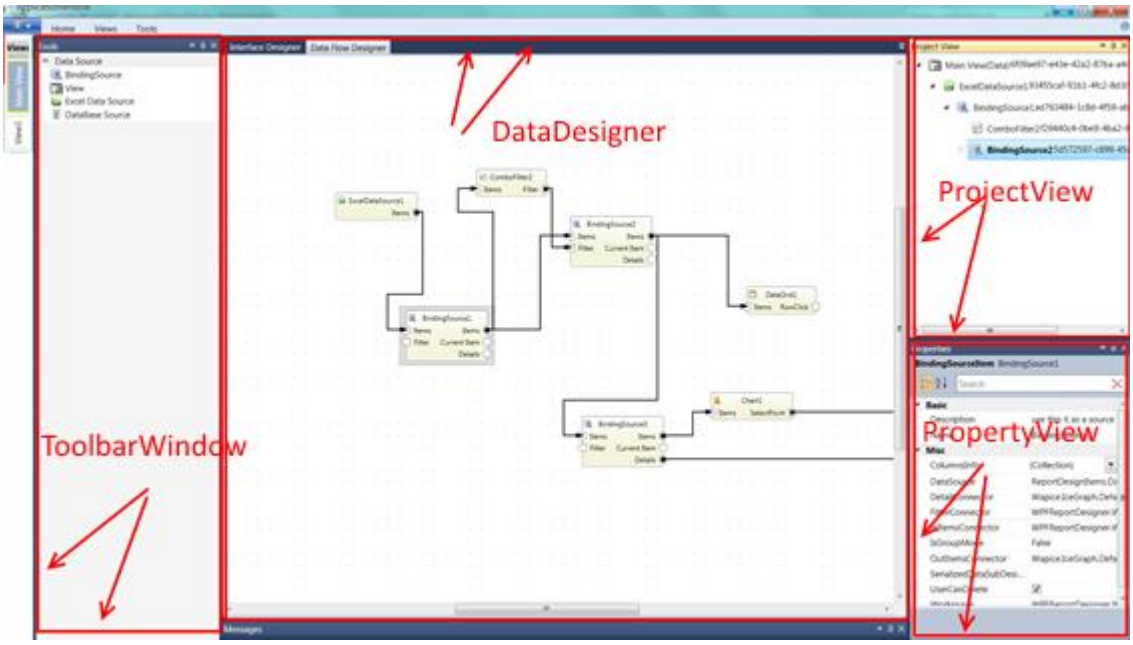

Figure 31 Data flow designer panel (report designer)

And finally the last screen shot is both UI designer and report designer beside each other.

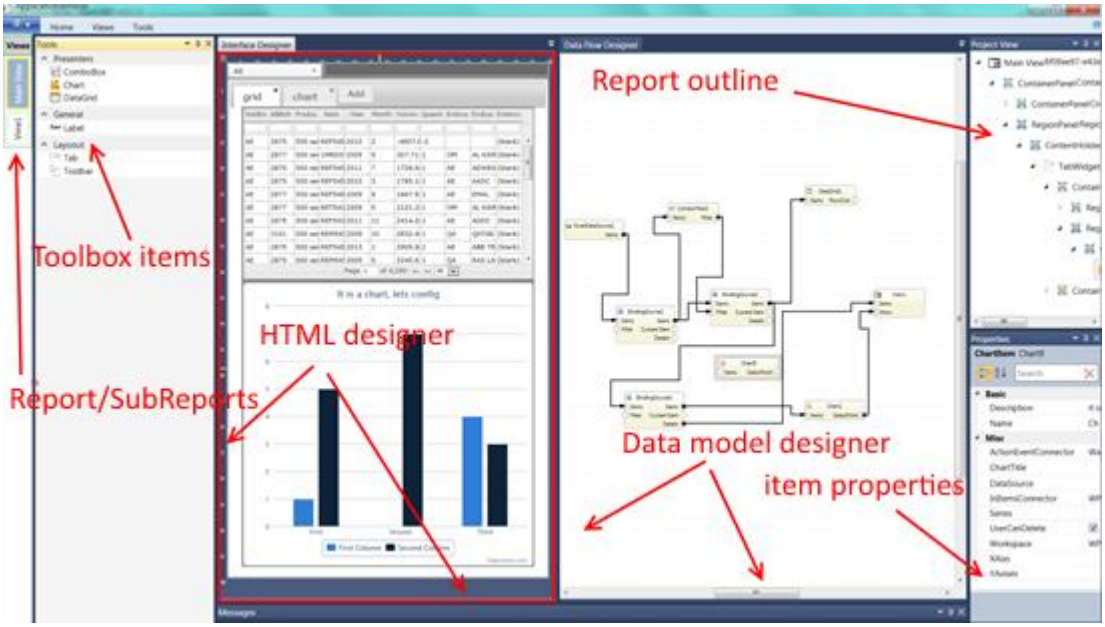

Figure 32 Both UI designer and data flow designer (report designer)

As it was explained there are two designer sheets to design reports one is *Data-FlowDesigner*; in this space users are able to select which data should be used and how that data should flow through UI elements and also how that data should be shaped for using in UI. Users can define and set data sources, customize data (add calculated columns, group data…), set bindings to UI elements and generally whatever is needed to design customized data flow. The second design space is an *HTMLDesigner* dedicated to design UI and UI elements and layout.

Users are able to design different views (sub reports) for their reports, it allows them to implement drilldown concept (for example designers are able to design a report showing a chart when its parts get clicked a new view with details of that chart will be shown).

Views are connected to each other (it is possible to navigate from one view to another), each view could be seen as a Form. The concept of *DesignSpace* is designed to support multiple views in a report. *DesignSpaceVM* class represents each view and by changing the active view (the active DesignSpace) in workspace, designer panels show information of that active view. DesignSpaceVM has an object for each designer panel an instance of HTMLViewVM which has information related to html designer and DataFlowViewVM is related to data flow designer.

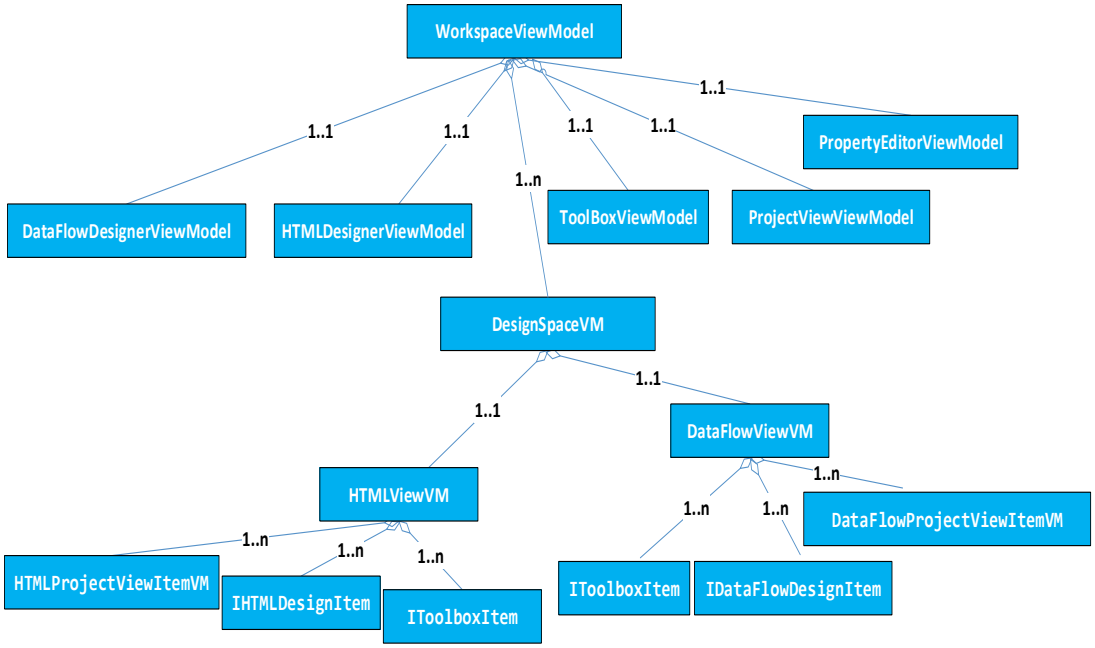

Figure 33 Main class diagram (report designer)

5.2.3.2 ReportDesignItems

In this project all of the toolbar items, report design items and related classes are located (items used to design reports in both HTML designer and data designer). There are two main namespaces, *DataDesigner* containing data design items and *HTMLDesigner* containing design items related to html design panel. Moreover, for the most of design items in *HtmlDesigner* or *DataFlowDesigner* (some might does not support toolbox item) there is a class implementing *IToolboxItem* or more specifically *IToolboxHTMLDesignItem* or *IToolboxDataDesignItem* these classes make toolbox items which will be added to the toolbox panel to allow inserting items to the designers. As a result if there is a design item called *XXX* there will be *XXXWidget/JsF* and *XXXToolBoxItem* under HTMLDesigner namespace and also *XXXItem* and *XXXToolBoxItem* classes under DataDesigner namespace. In Figure 34 HTML toolbox items, data tool box items and their related panels are shown.

In Figure 34 relations between *IToolboxItem*, *IDesignItemBase* and the other inherited interfaces are shown.

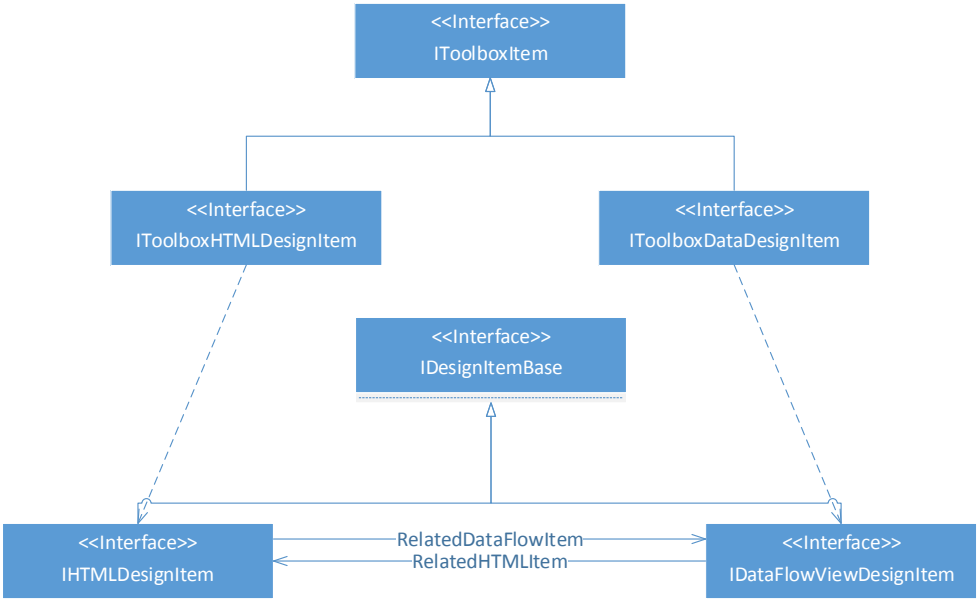

Figure 34 IToolboxItem, IDesignItemBase and related interfaces class diagram

To design a report, items are living in three different spaces html designer in C# space, data designer again in C# space and finally HTML report page in JavaScript space. Each design object is composed of at most three connected different parts in these three contexts: an instance of a C# class to manage the visual part of the design object in C# space which is implementing *IHTMLDesignItem*, a C# class representing the item's data part used in data flow designer implementing *IDataFlowDesignItem* (these two related classes are connected with a property) and finally a *JavaScript* class to represent the concept of design item in HTML/JavaScript space. Figure 35 shows the class diagram of design panels and classes related to a design item. The green boxes are three different classes for a design item and blue boxes showing their container space the whole diagram divided by two gray boxes which showing .net and JavaScript space.

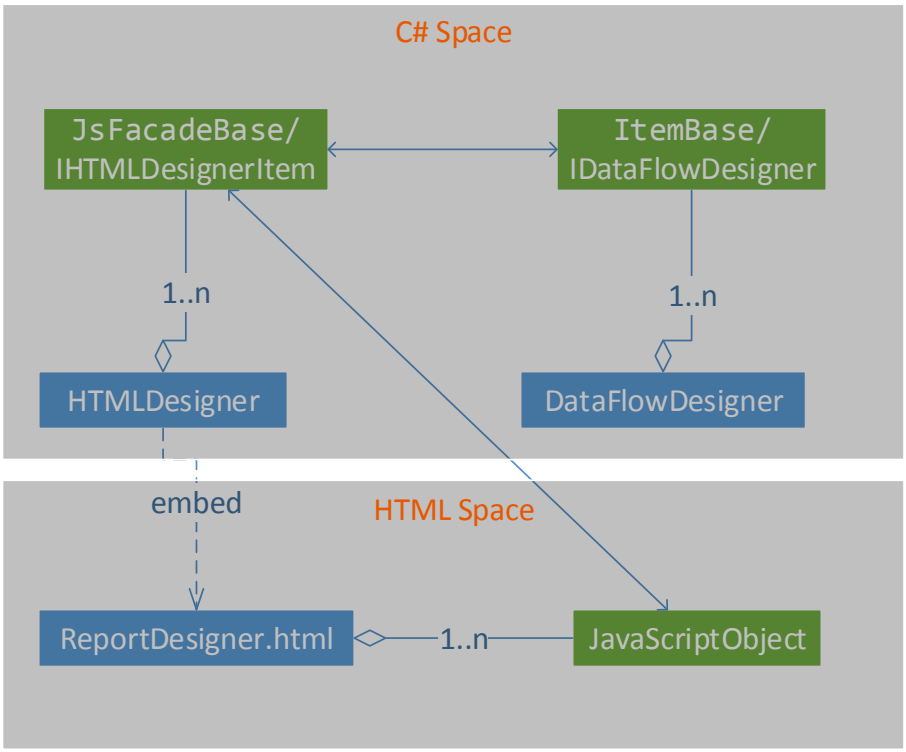

Figure 35 Design items class diagram

In this project there are two important namespace defining classes related to each UI designer or data designer.

5.2.4 JavaScript and HTML project

JavaScript and HTML files are physically located in *ReportWebServer* project. All the JavaScript and HTML files related to three different pages (*ReportDesigner.html*, *ReportViewer.html* and *ReportBrowser.html*) are in the same project and some classes are used for different scenarios as a result there might be some functions in a class that are used in some scenarios but not in another scenario. Also, because design items are used in hosted scenarios, there are some methods which get invoke from the host application as well. It was tried to follow prototype pattern to implement JavaScript classes (mainly).

- **HTMLView JavaScript class**

JavaScript application is composed of multiple views which are instances of *HTMLView* implementing the main report and sub reports. *HTMLView* is like a Form with some items or it can be seen as a report and sub reports in *HTML/JavaScript* context. Based on the needs users are able to add a view (sub report) then they are able to send special data to a view (from main data flow in parent view), also users are able to set *show* event of a view to an action like a chart or a data grid row click event.

To make it clear in design time instances of *HTMLView* could be seen as different Form types used to design an application (Forms in design time) and in run time instances of *HTMLView* could be seen as instances of those Forms(designing a form in design time and using instances of that form in run time).

- **JavaScript DesignItems**

In Figure 36 the class structure for JavaScript design items is shown and form this diagram it can be seen how different JavaScript classes for different design items are related to each other.

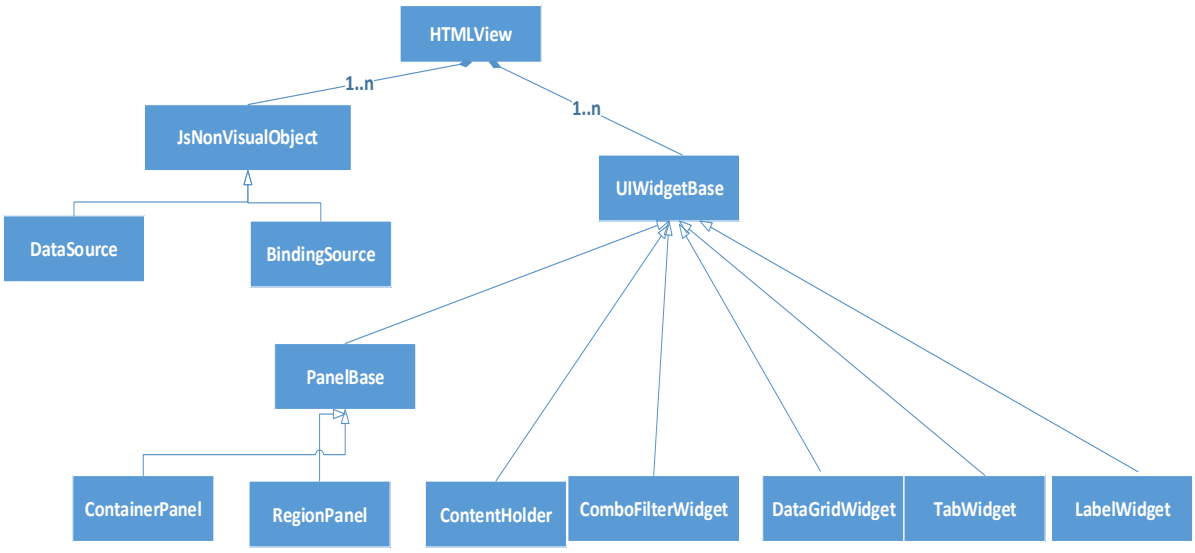

Figure 36 JavaScript class diagram

There are three classes which manage the layout of a report: *ContainerPanel, RegionPanel, and ContentHolder*. Each container panel can have different *RegionPanels* or another *ContainerPanel*. In Figure 37 the relation between different panel classes can be seen.

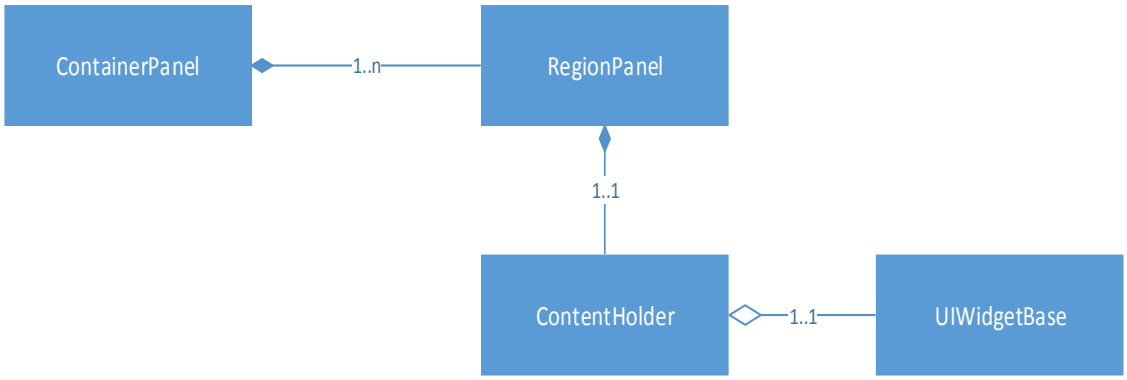

Figure 37 JavaScript layer and panels' structure

5.2.5 Other Modules

There are other projects in the software solution which play complementary roles here a short explanation of them is mentioned.

- **Common**
  This project has classes and interfaces which are shared between client (*WPFReportViewer* and *WPFReportDesigner*) and server, it contains service interfaces and entity model objects.

- **Tools.Cmn**
  This project contains general utility classes which are used in the solution like .net extensions and etc. (not *WPF/UI* utility classes)
- **Tools.WPF**
  This project contains basic *WPF/UI* utility classes.
- **AppResources**
  All the images and WPF resource dictionaries are located in this project.

6 EVALUATION AND DISCUSSION OF THE PRODUCT

In this chapter the result application from technical and usability points of view is discussed. In the first section, selected development technology as the most challenging decision is mentioned and the problems and difficulties regarding using HTML and JavaScript as development platform are discussed. In the next section of this chapter, first a holistic view of the usage workflow of the report designer application is explained after that a usability inspection of the UI and workflows of the report designer application is mentioned, in this usability analysis it was tried to compare result product by its counterparts and discuss its UI model and features causing problems for users; problems regarding to speed, flexibility and efficiency of use, complexity and distractions in designing reports and analyzing data. Also the amount of user control and freedom, aesthetic and minimalistic design, visibility of the system status and the amount system helps users to get to their job done, as heuristics and measures for evaluation of the usability of the product are used.

6.1 HTML and JavaScript as an Implementation Platform

Selecting a technology to develop an application is one of the most important design decisions in early stage of software development process. There are many software development environments and languages each one has its own positive and negative properties. Especially nowadays there are different devices and operating systems and each development environment supports some run time environments better, so it is hard to choose a technology to be able cover the most platforms (in the best way).

HTML (5)/JavaScript is an exception, it can cover almost all the different devices and operating systems. Specially after introducing *HTML5* and after growth in the level of HTML5 coverage by different browsers and availability of rich *JavaScript* open source libraries like *jQuery, angularjs, requirerjs* and etc. also more supports for *Ajax* calls and other application development facilities in *HTML5*, developing *HTML5/JavaScript* based applications in browsers got more popular.

In the past mainly just static web applications were done in HTML beside a little bit client side code in the browser (to support validation or to give some trivial client side feedbacks), but recently improvements in HTML and new features of HTML5 caused the concept of Single Page Application (SPA) be introduced by modern web applications. SPAs are an attempt to bring user experience of desktop applications into applications running in web browsers by the help of modern features of HTML5.

The spirit of a *SPA* is putting UI logic in the client side and freeing server from managing UI logics and UI states which results in a good users experience close to desktop

by reducing server calls (which might be slow), less dependence on the server (which might be busy doing something else), also by caching data and Ajax background calls. In SPAs servers have the responsibly to generate just the first page of the application and after that all UI rendering logic will happen in the client and server just answers to data and resource retrieval calls from client.

The report viewer should be executable in all the devices and platforms (Mobile BI requirement) and it would be a big advantage (comparing with competitors) to have a designers just right in the browser, by this developer team can produce just one version of the application and it can be run with the same look and feel in different platforms. Besides, the concept of *SPA* for this product quite fits, lots of UI logic in the client and server should just answer to data retrieval calls. So as it was explained in the third chapter *HTML5/JavaScript* was chosen for future releases and it was decided to have a hybrid html/native (window desktop) based solution for the version one (mainly in HTLM/JavaScript but an application loader in window desktop format).

HTML5/JavaScript is portable it is a really important feature though it is not the whole story. Using JavaScript to develop big applications is a big risk for projects and development teams (middle size teams and with limited resources). Although there are lots of good libraries and tools to manage JavaScript applications, it is a really hard job to harness JavaScript behaviors in enterprise applications. *JavaScript* and *HTML5* is not still ready to be used to develop big applications (at least for companies who do not have enough resources to develop their own custom tools), there is not a good *IDE* to work with *JavaScript*, and *JavaScript* in its current version does not support object oriented programming inherently (though there are some methods and patterns to implement *OOP* in *JavaScript*). There are different open source libraries to help *JavaScript* application development but at the same time it is a complicated task to select the best libraries. Generally it can be said, JavaScript codes are not maintainable and extendable and using it in development really hinders software development process (which slows down the development process).

Beside technical difficulty in developing applications with JavaScript, the second important problem using *JavaScript* and *Html* in this project was providing flexibility in UI of report designer. It is really important for report designer (more than report viewer) to support users in visual interaction with application components (in some areas like managing mouse events, resizing, drag and drop, handling precise location of controls , supporting redo/undo actions, showing modal dialog boxes, handling async operations, auto saving of user interface layout and also error prevention and error recovery and etc.), *HTML5/JavaScript* is not flexible enough to support advanced actions, and in the case of this product it limited the ability of programmers in developing a smooth experience for report designer.

After finishing the first version of the product, it seems choosing *HTML/JavaScript* based technology for this prototype version was not a good choice especially hosting *WebBrowser* control inside *WPF* application was like a hassle. This choice really slowed

down the application development process and added lots useless pressure on development team.

If this version was intended to be an executable prototype (which it was) it would have been better if it was developed fully with WPF even at the end if the team would decide to develop the main application with *JavaScript* and have to throw away this prototype, still speeding up this phase was a good reason to develop the first version fully with WPF. Using a rich environment like *WPF* could have speeded up the prototype version development process and at the same time development team were able to find the complexity of the application, evaluating the important usability concerns (interaction and visualization techniques) also producing something presentable for customers.

6.2 Usage of the Product and Issues

One of the most important success factor of an SSBI tool is supporting users with good a usability and experience while they are interacting with visualization and data in both report designing and report viewing and analyzing data. A short look at the current market of SSBI tools proves that big SSBI vendors are investing heavily on researches (and already implementing) to find intuitive, innovative and easy to use data visualization and interaction techniques to be able to dominate this market.

Conducting a formal usability evaluation needs quite extensive resources and time (itself could be a topic of a thesis) here we just did an **expert evaluation** of the product usability and it seems this kind of usability evaluation at this step of the product is quite effective way in pointing problematic parts of the application to help development team for next versions.

The most complicated part of the product is report designer (techniques in report viewer is somehow are derived by main techniques coming from report designer) in this section produced application from usability point of view, in designing reports, interacting with data and visualization is discussed. In the below sections, first an example of the application usage for designing a report from users point of view is explained (to explain the workflow that users have to follow) and in the next section application workflow and its UI are evaluated based on some usability factors and practices.

6.2.1 Report Designer UI and Workflows

Each tool that support users in performing a task has some steps and users should follow those steps to use that tool. In Figure 38 users actions needed to produce a report by the application are shown. These actions will be done in two separate areas data designer and report designer. Users will use data designer to design data flow, report items' behaviors and data modeling, on the other side, they will use UI designer panel to design the look. In this section this workflow is described in more details.

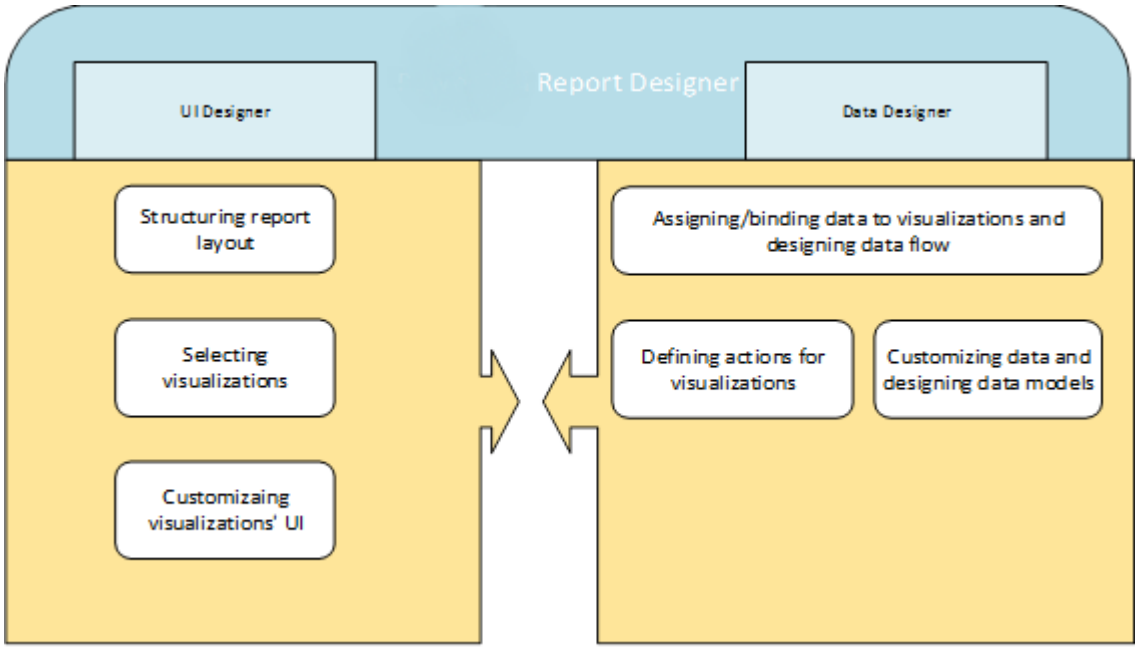

Figure 38 Report designer activities

Figure 38 shows that users should do different activities in both UI designer and Data designer and also it is listing activities are done in each space of Data designer and UI designer.

Now let's have a short sample of designing a report and analyzing some data with the application to able to understand activities needed to be done by users (report designers). Then we will be able to evaluate the amount of work and difficulties of tasks while using the application.

I imaging we have an Excel file containing some sample data of an imaginary company selling some products in different part of a country. The dataset contains records of each occurrence of sale containing the name of the seller, item, unit and the total price. We want to design a dynamic report for this data and at the same time analyze company's profit by the report designer application.

First we want to have a data grid showing the list of sales and also we want to be able to filter this list by the name of items (designing a report that dynamically could be filtered). So (after structuring report by adding a *TabControl*) we put a *DataGrid* and a *ComboBox* on UI designer. Then we switch to data designer, put an *ExcelDataSource* in designer and connect that to our excel data set containing our sample data, next we draw an edge from *ExcelDataSource* node to a *BindingSource* node and then another edge from *BindingSource* to the *DataGrid* node, by these actions the data will be shown in our data gird. The next step is showing items' list in the *ComboBox*, for this we draw a line from *ExcelDataSource* node to the *ComboBox* node then we draw a line from *ComboBox* filter action to the *BindingSource* filter event, so when users select a record in the *ComboBox*, items in *BindingSource* and indirectly items in the *DataGrid* will get filter. In Figure 39 you can see the state of data designer panel.

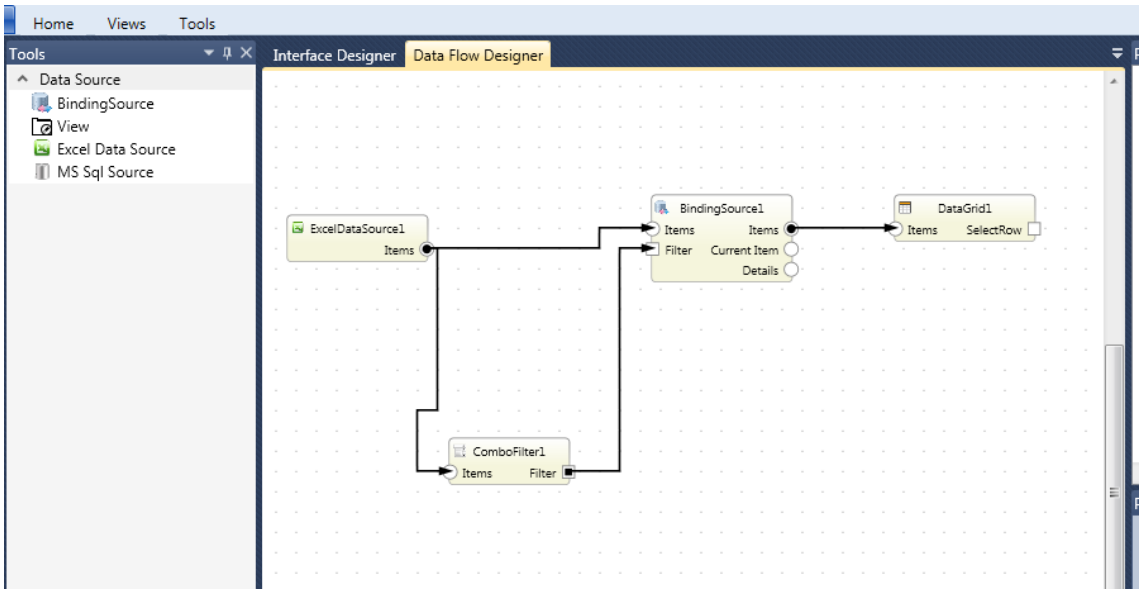

Figure 39 Data designer (retrieving and showing data)

And the next screen shot is showing the state of UI designer after the above actions (which is what we wanted to design).

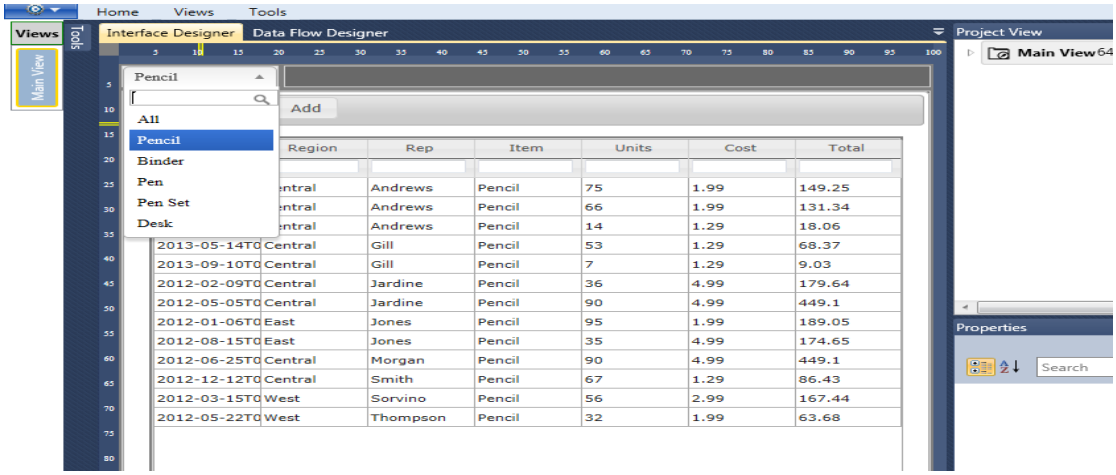

Figure 40 UI designer for retrieving and showing data

Now imaging we want to add a new column (calculated column) in our dataset to show the pure profit which imaginary can be calculated by below the formula:

If **unit** was **equal** or **less** than **50** then **PureProfit** is **0.5 * Total**
If **unit** was **more** than **50** then **PureProfit** is **0.7 * Total**

To add this column we should edit *BindingSource1* and add a new calculated column as below.

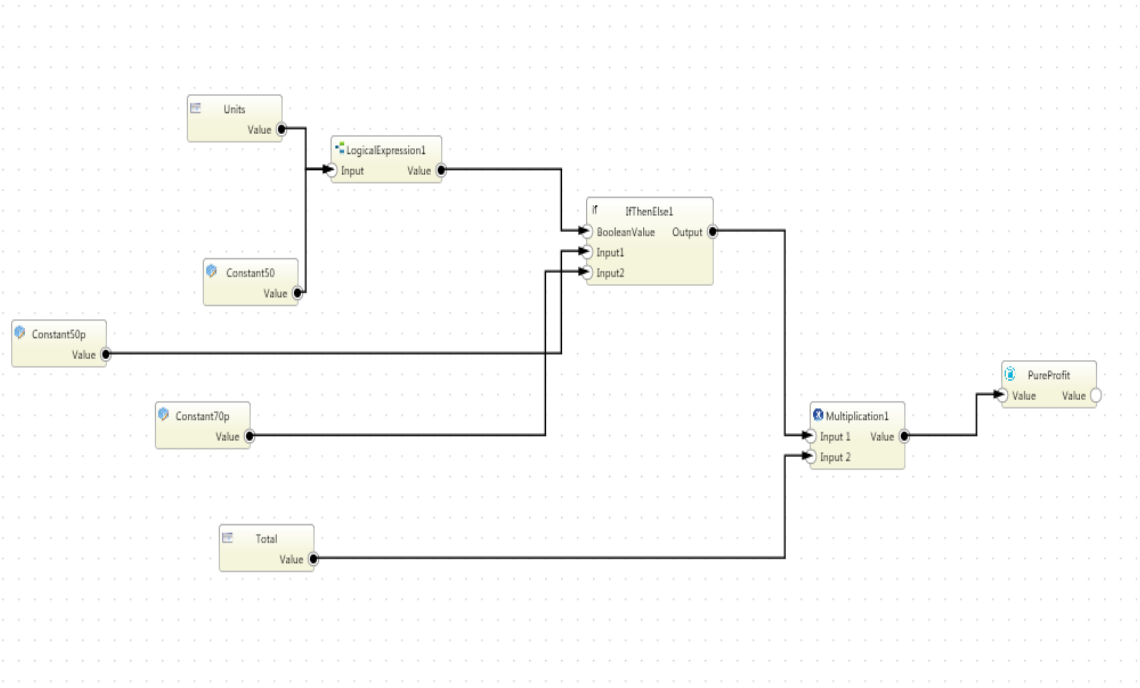

Figure 41 Data designer (adding business logic)

From Figure 41 it is visible one *logic* node, one *if-then-else* node, three constant nodes, two *columns* nodes and one calculation node plus some *lines* are used to design this small business logic for adding a new calculated column. In Figure 42 the result in the data grid is visible.

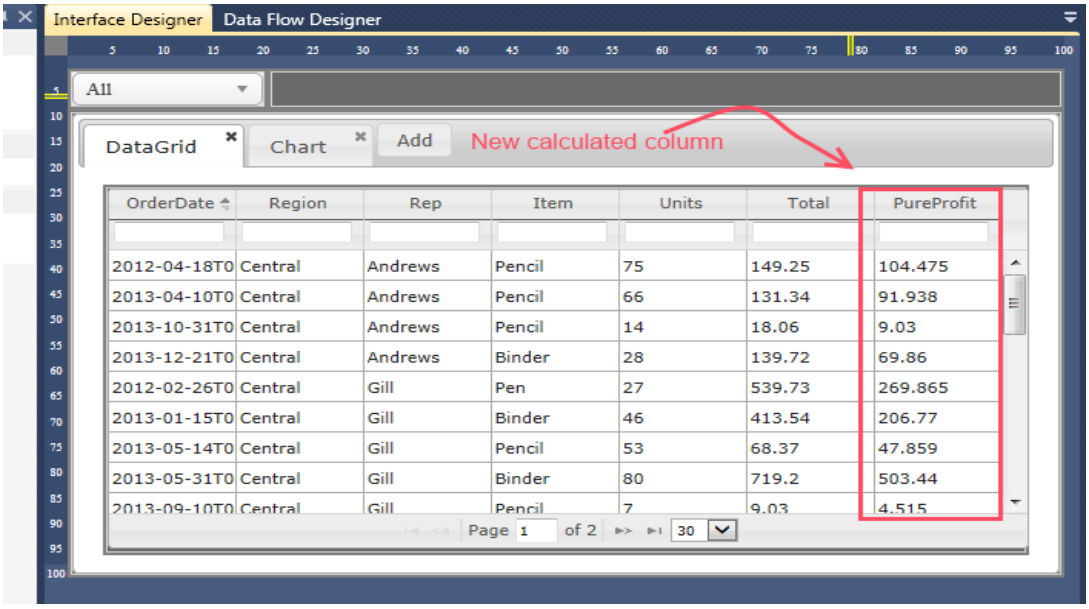

Figure 42 UI designer the ne calculated column in data grid

To complete our design lets analyze our data by grouping our sales records by **region** field and see the total amount of sales in each region. First we should add a chart control in the UI designer then we go to the data designer and add one more *BindingSource* to get data from the main data source and set it into the chart.

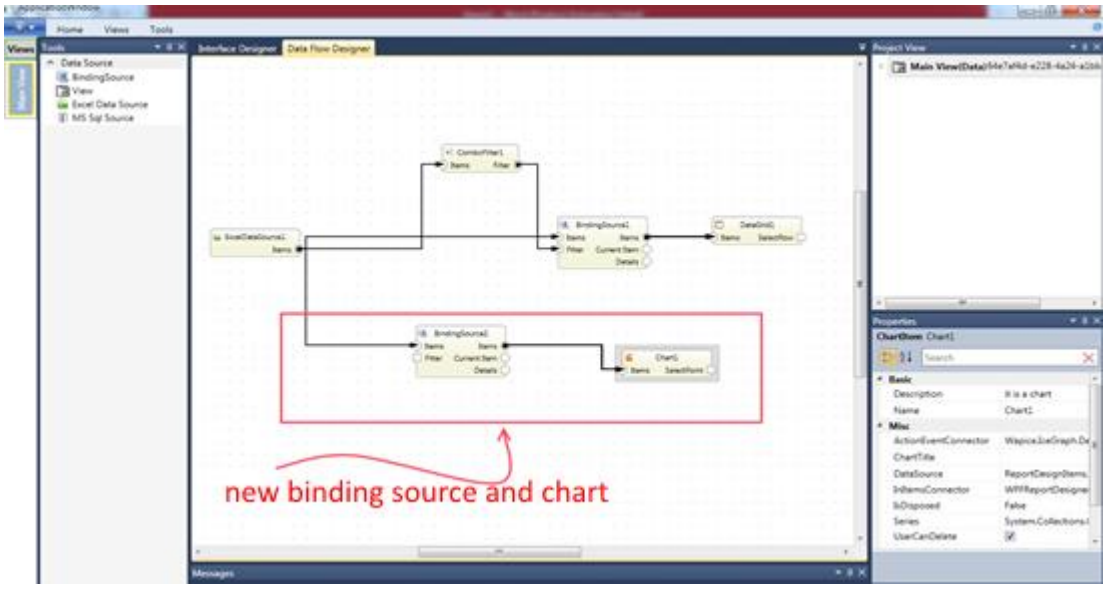

Figure 43 Data designer (grouping data)

We should now edit our new *BindingSource* and group data by region field and add an aggregated column to have the sum of total price in each region (something like Figure 44diagram).

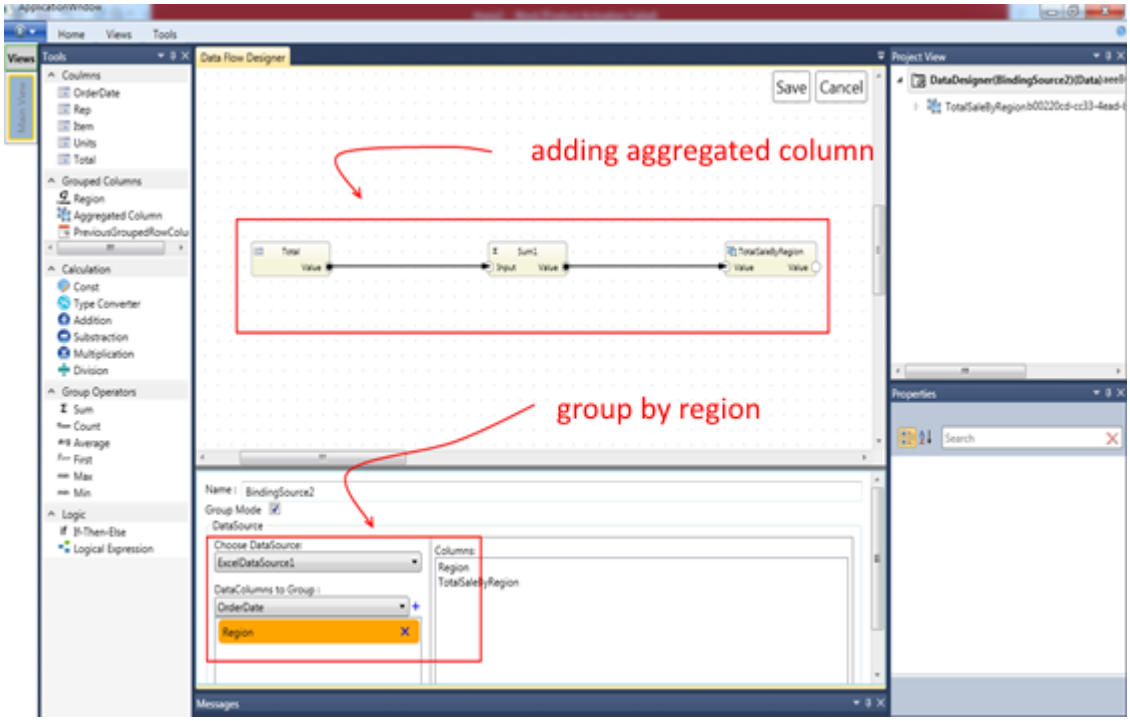

Figure 44 Data designer (adding aggregated column to binding source)

After doing above steps and selecting the chart as a pie chart, we will have the below result (Figure 45). The total amount of sale in each region is shown as a part of a pie chart.

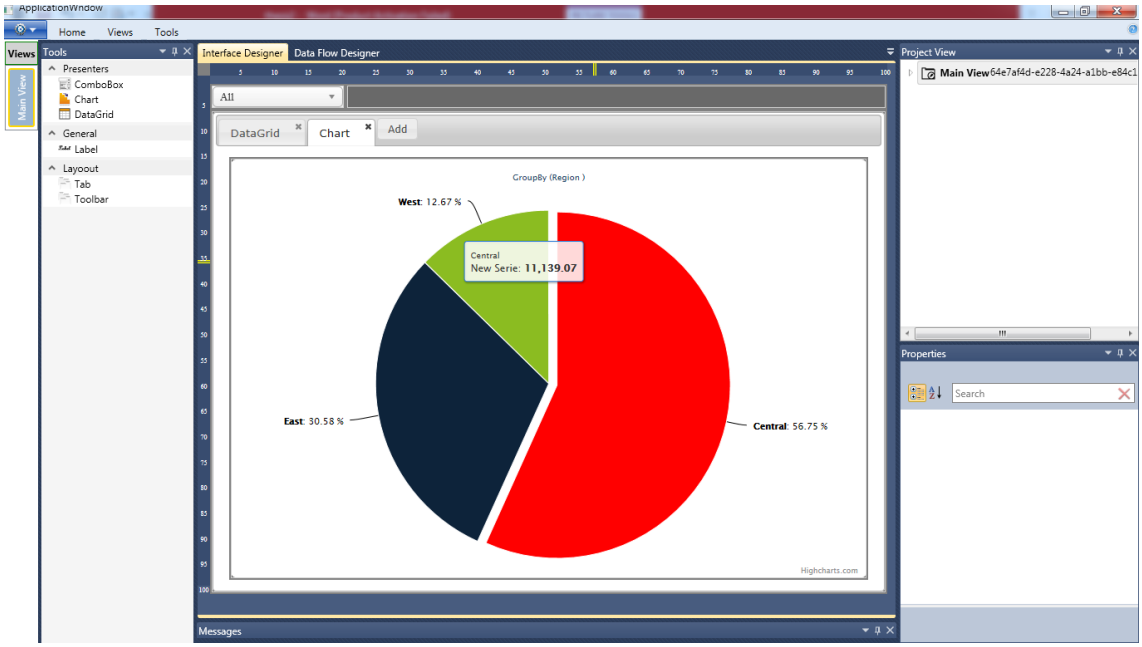

Figure 45 Pie chart visualizing grouped data by region

Finally we want to add drilling capacity to our dynamic report to get more insight of our data (by this, we can look at the amount of our product sale based on the region and also we can go to details and see the amount of sale in a certain region divided by items). So we have to design a sub report to connect *click* action of the *chart* node to *show* event of the *view* node and also get detail of data from *BindingSource*2 (grouped binding source) and set that data into our view node.

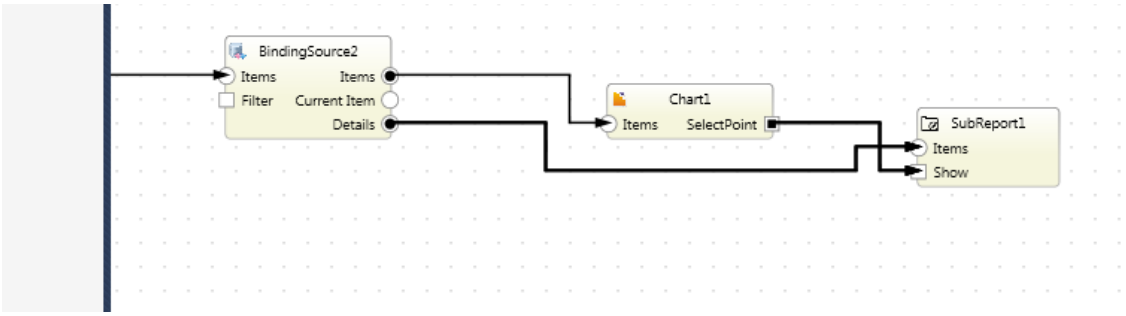

Figure 46 Data designer (adding a view node)

By this if users click in each region of the pie chart a new view called *SupReport1* will be shown and details of that selected data will be passed to that view. So we have to design *SupReport1* the same way as this main view. So we add a new chart in sub report UI designer then, in data designer of sub report we add a new binding source to get details of data which are passed to this sub report, group that data by *item* column then show that data in our new chart.

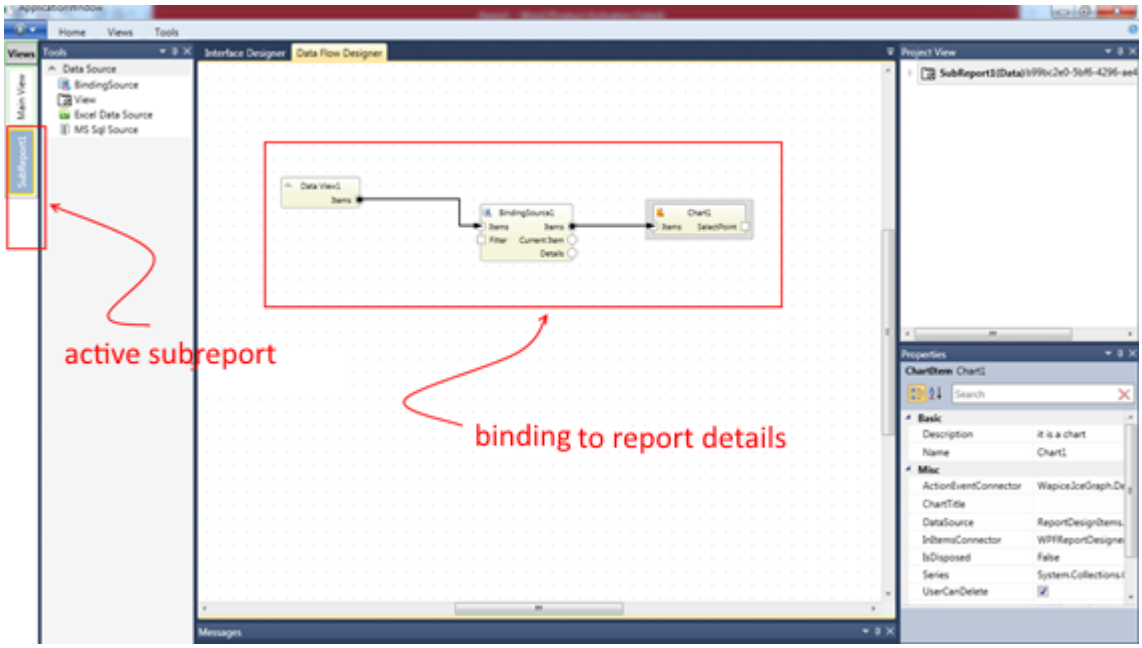

Figure 47 Getting data and show in the chart

And the next screen shot is showing what we have done to get data grouped and making new aggregated column for our new chart.

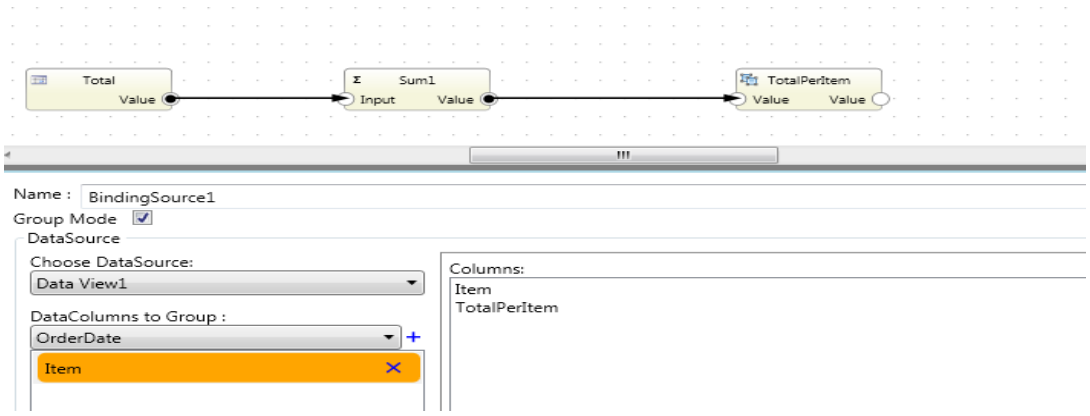

Figure 48 Data designer (grouping data by Item and making aggregated column)

Finally by doing the above UI and data design actives we will have a chart in our main report in which our users by clicking on a region like *Central* region, can drill into details and see the details of sale for that region grouped by items. This drilling could happen in any level (drill of drill of drill…) and gives a powerful way of designing sub reports and analyzing data. In Figure 49 this final view (when we drilled to the details of Central region sales data) is shown.

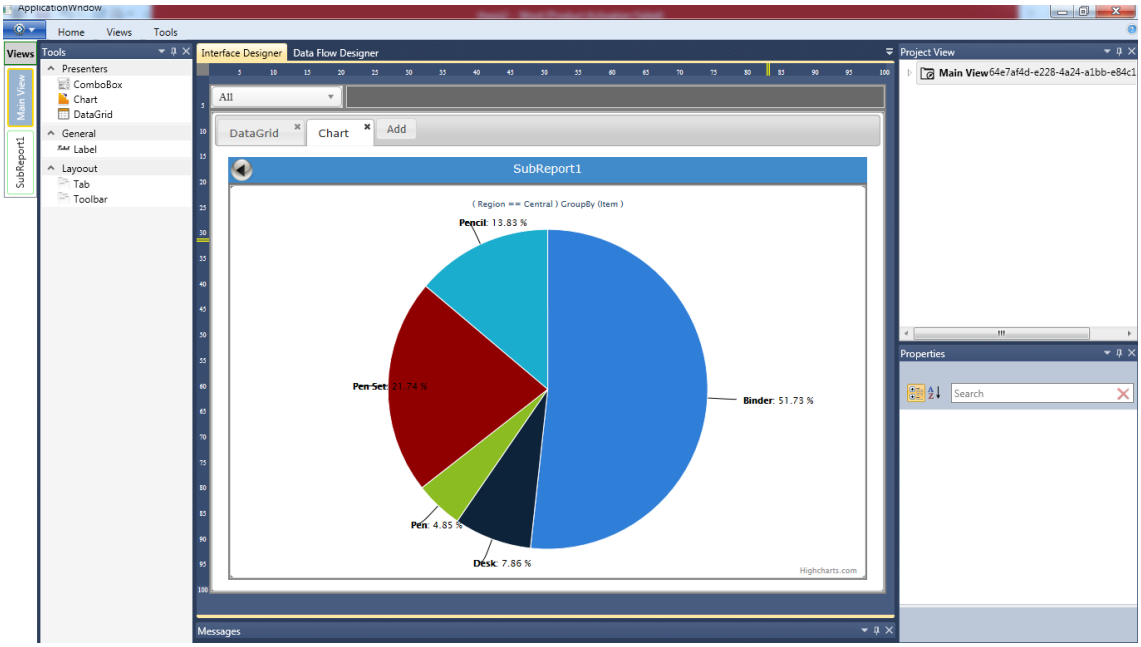

Figure 49 Drilled pie chart showing total sail by region by item.

6.2.2 Evaluation of the Product Usability

As it was discussed in chapter one, usability is the extent of effectiveness, efficiency and satisfaction by which users can get their job done using a product. The main task of the report designer application is supporting users in producing reports and analyzing data.

In this section we will discuss the product workflow model (in some cases in comparison with other models) and UI model features causing problems for users; problems regarding to speed, flexibility and efficiency of use, complexity and distractions in designing reports and analyzing data, also we will talk about the amount of user control and freedom, aesthetic and minimalistic design, visibility of the system status and the amount system helps users to get to their job done. This is a short list of usability measures we used in this section.

6.2.2.1 Two Design Space Areas

As it what was explained above, users will draw the look and layout of their report in UI designer panel, and in the data designer panel they design the behaviors, data flow and data models. For (almost) each UI component in data designer there is a data node in data flow designer which users use that node to customize data for that UI component.

This model has its own powerful features, though has a big problem, report designers have to cope with the complexity of managing two panels. Each designing area has different focus (different structural concepts) so users have to change their mind set coming from one designer panel to another designer panel (which it happens quite regularly). Besides, managing two entities for each visualization (like a Chart) is another drawback of this model of interface. In the UI designer because items are locating inside each other

hierarchically it is not that hard to find a control but locating a node in data designer is a hard job.

As a result, it seems this conceptual design of dividing designing space into two separate parts not just slows down the report designing process, it forces users to concentrate and manage two different spaces at the same time resulting in a complicated and distracting report designing experience.

Generally it can be said, considering report layout designing as a separate activity is might give a powerful ability to users to be able to control the look of their report in details (it supports users control and freedom heuristic principle), but has a impact on simplicity of report designing resulted in a big drawback which is reducing the speed and efficiency of report designing experience.

6.2.2.2 UI Designer: Designing UI Forms or Dashboards

The next concept close to the previous one (even can solve the previous problem) is that it should be cleared what is expected from report design process a **Form** based visual report or a **Dashboard** based report? Based on the current arrangement of UI designer, users will be tempted to design a full featured **Form** like users interfaces for their reports. The mindset behind designing this application model was based on the legacy report designers (some close concepts with user interface programming environment which are composed of a visual Form designer and code behind, in technologies like Windows Form, Java Swing and etc.), in this model, each report (implicitly each sub report) modeled like a Form in which users are able to design their reports by drag and drop UI components from a tool box into the UI designer (Form). Users are able to divide their reports by layout components like group box, tables, tab controls and etc. after giving a report a structure or layout then report designer will add other visualizing controls like Charts, DataGrids and etc. to show data. This workflow model for designing a report gives users a quite high level of freedom in controlling the layout of their report; they can shape their report however they like. This model is quite accurate and understandable especially for users with technical programming or report designing background (it is quite like all legacy report design tools).

Though, having a quick look in new (dynamic) reports seems new dashboardish reports got more attractive, engaging and popular to people who want to browse designed reports. Moreover, it is much easier and simpler (without needing a time consuming layout design process) to design them. Usually dashboard based reports are built based on few related charts and filtering components (few visualization components, few actions, rich filtering and a simple layout) results in nicer reports with less efforts (minimalistic).

This application supports complicated scenarios in designing reports, but this unnecessary UI layout structuring facilities has changed the model of report creation. Now users have to do a comprehensive report design process and at the same time the result might not be as attractive as simple dashboard based reports..

6.2.2.3 Report Items, Creation and Editing Visualizations

As it was described above, users first design the UI of a report by dragging a control on UI designer and locate their visualization item, then they will assign data to report controls in data designer panel. Although this designing workflow might be familiar for UI developers and technical report designers, it seems double works, first design UI then assigning data slowing down report design and analyze experience for report designer users.

In modern SSBI/BI tool a different and shorter workflow is used for designing reports. In this new model (can be found for example in *MS PowerView* and *Tableau*) having a visualization happens just with one action. Users drag some data (and not UI control) into designer space and then a default visualization based on the type of data will be generated in the designer and users can change this default visualization if it is needed. For example users can drag a list of data items into the designer and a list view will be generated if it was needed users can change this visualization (for example by a context menu) and convert the list view to a chart. This short workflow allows users to be able to play with data easily and do their analyses faster (add visualization, remove it, and change it rapidly).

Now if we compare this simple workflow with our product, and think of a situation in which users add a chart control inside their UI designer then they want to change it to a data grid. In our application and model users have to delete the first chart and put a data grid in the place of that chart then again assign data to that new data grid which is lots of work.

Figure 50 is one screen shot from *Tableau* software; the *Column* chart is selected, but if we want to change the visualization we easily can select another visualization from the right tool bar (contracting with this product in which users have to remove the previous chart and make new visualization from ground which is too much work.).

Again it can be easily seen that implemented report design model (how users should use UI components to design their reports) does not support users' freedom, flexibility and efficiency in adding, removing and editing visualizations and slows down the process of report designing.

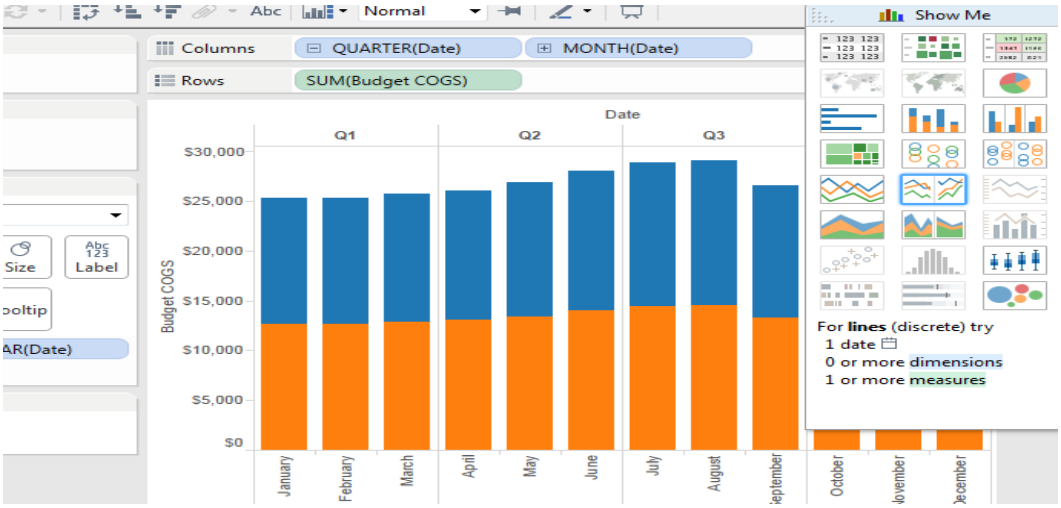

Figure 50 Tableau screen shot (changing visualization by a mouse click)

6.2.2.4 Using Intelligent Interface to Help Users in Analyzing data

The new trend in SSBI/BI tools (generally in any modern user interface) is using hysterics and artificial intelligence in guiding users to gain what they want. New interfaces are intelligent enough to help users to reach to their goals easier and faster. In SSBI world, *Q&A* of *MS PowerBI* is one of the best examples of these interfaces, users just ask a question about data in human understandable way and application will draw the best visualization which match the spirit of that data (based on the type of data, amount of data and etc.) for users. Another sample is visualizing a dataset which contains some geo information, the SSBI tool will understand this data type and will draw a geo map for that data.

The other usage of this intelligence in SSBI coming in sense of data interpretation by SSBI tools, if two datasets somehow be related through their fields, tool automatically enables drilling for that visualization (without needing it explicitly be designed by users which results in a faster report design and data analyze experience). For example in modeling of a data set, tool somehow might understand in each data record there is month, year and a measure value (like the amount of sale) so if the sum of this measure is shown in a pie chart grouped by year, tool will infer that it can show this sum of data in each month of a year as well, as a result it will add the ability to drill inside the data of each year automatically, then users by clicking the main chart can go to the details of that chart getting a view of the sum of that value grouped by each month in the selected year.

This report designer application needs a big improvement in this part to provide a smoother report analyzing experience for its users. It should has some features to help its users to recognize, diagnose, and recover from errors in designing data models and be more efficient in repeatable tasks.

6.2.2.5 Drawing Graph to Design data Model

Any BI/SSBI tool should give the ability of managing data and customizing data models by some user interface interactions to its users. In this application we have two levels of data design, in first level users design data flow in the report and in the second level they can reshape data or build a new data model by customizing a BindingSource.

The data modeling in this application is based on piping/wiring data or connecting nodes to each other like a graph. These kinds of interface are quite popular in **workflow designing,** each node represents a task and connecting nodes to each other will shape the workflow. Even before visual workflow designers, this concept was being used in visual programming and visual programming languages.

On the one hand, this kind of modeling logic (with help of nodes and edges) is really powerful (as it was shown in the above section, by this method users are able to design any kind of logic by drag and drop), on the other hand it is time consuming interaction and when the designed logic gets bigger it results in a complicated networks of logical nodes hard to manage.

This kind of node, edge logic designer could be compared with formula editor which is quite popular in different tools used to get input script from users. Formula editors are simply a text box which users can both write script (usually with a simple if then else logic language) also they can interactively select building blocks of formulas by mouse click from a menu.

Now let's have short comparison between designing the same logic in this application graph based designer and also in a formula editor designer from **Omniscope** [51] data visualization tool, then it is possible to see the amount of work and complexity between these two user interfaces.

Imagine a user wants to add a *PureProfit* calculated column by below logic to his current data set:

If **unit** was **equal** or **less** than **50** then **PureProfit** is **0.5 * Total**

If **unit** was **equal** or **more** than **50** then **PureProfit** is **0.7 * Total**

Figure 51 is showing from formula editor of *Omniscope tool.*

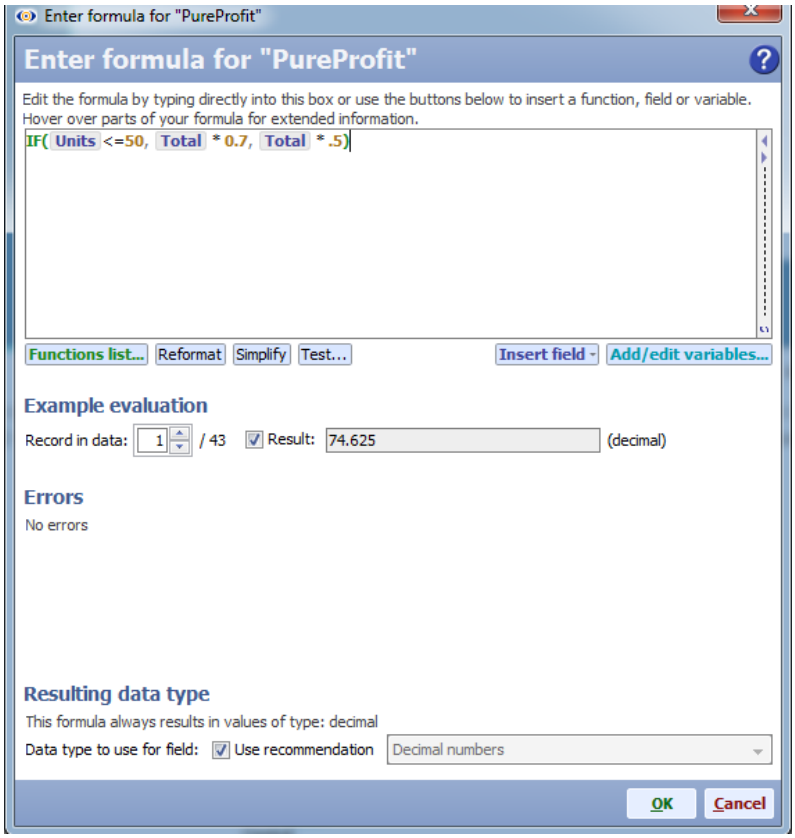

*Figure 51 Data logic in formula editor (*Omniscope tool*)*

And the next one is the same logic implemented by node, edge concept in our report designer.

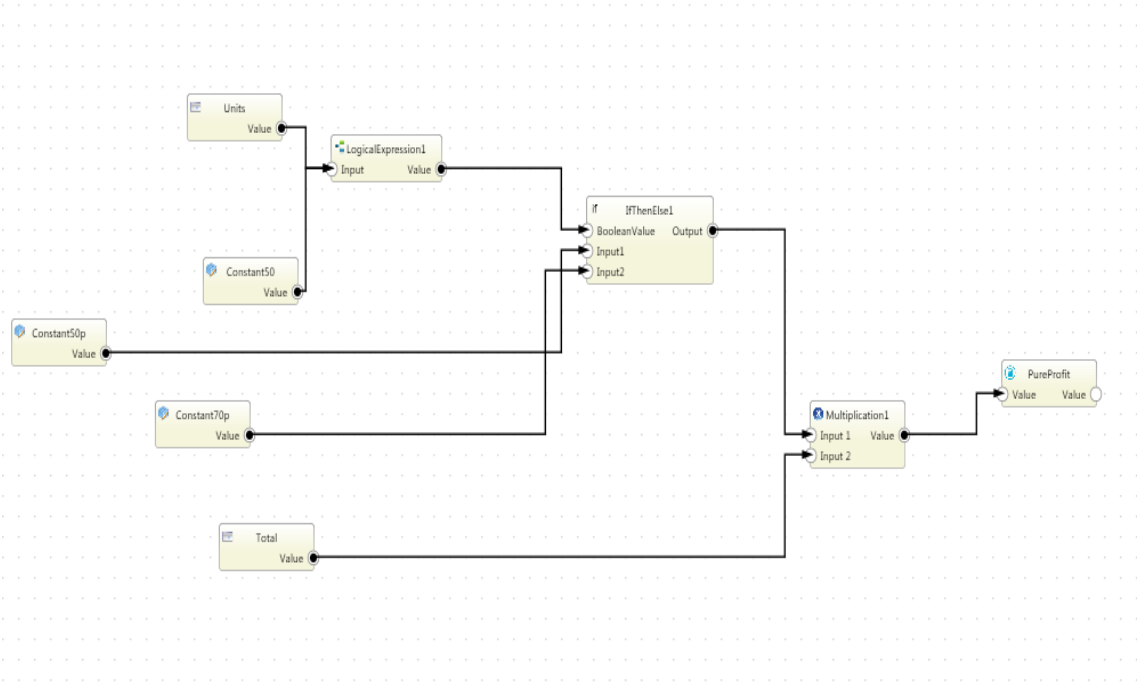

Figure 52 Data logic in node, edge data flow designer

It is clearly obvious in this application users need to do much more interactions than a simple formula editor to add the same logic. Even if the user wanted to select data fields from the list (and not by writing their names) the text editor at most needs 5 actions to

perform: one for if-else-then function selection, three for selecting fields and one for completing text formula but in this application, users should draw several nodes and edges.

One problem here is that, the concept of designing a workflow is quite different than designing data models, in modeling a workflow usually the graphical view itself is important for users to understand the whole task (the workflow visualization itself is a valuable documentation diagram) but in designing data this visualization is something one time usage and users do not refer to it again that much.

It can be said, this kind of data designer with nodes, edges and creating a graph of logical nodes is hard for understanding by novice users and complicated to manage for power users. If the main target of this interface are power users with a good technical background then why not providing them an easy to use, neat and concise formula editor interface and on the other if the users are not technical people this user interface is still hard for them, so this interface cannot bring a good usability neither for power nor novice users.

7 CONCLUSION

It took almost one year to have the first version of this product be ready (this current thesis work). It was a complicated application and development team encountered several technical and usability issues to design and develop, and too many details to cope with, and finally at the end an executable application was ready for piloting purposes. One important outcome of this development process was understanding of the scope of the application and finding a realistic estimation about the resources needed to develop a full feature SSBI tool.

As it was discussed in the previous chapter choosing HTML/JavaScript environment caused technical difficulties and slowed down the development process. Still seems JavaScript and HTML5 is not ready for rich UI applications like report designers. So in the case of this product, choosing this technology for product development, imposed more resources and difficulties in developing this application.

First of all, JavaScript as a language is flexible to support developers with different levels of expertise (novice programmers can easily use it to develop web applications), but it is not really built to be used for developing complicated and big applications. JavaScript is more a functional language than an object oriented language, so object oriented concept and practices like modules, classes and inheritance and etc. - which allows developing maintainable source codes - are not something inherently supported by the language (though there are some practices and patterns to apply object oriented concepts in JavaScript). The second problem of developing application in this environment is the lack of a full featured integrated development environment (IDE) for writing applications in JavaScript and debugging them which is mainly because JavaScript is a dynamic language not supporting static typing. The third big problem of developing HTML/Browser based GUI is about difficulties in controlling HTML components when they are hosted in a browser. Generally native solutions have more accurate and predictable application life cycle and UI component's behaviors than web applications hosted in browsers. For instance if we think of memory management, event systems (for example the sequence of drag and drop event or etc.), using shortcut keys, flexibility of UI components, GUI speed and etc. there are many quirks and twists to handle while using HTML GUI which impose the needs of significant resources and quite much time to test and solve them.

From usability point of view, one remarkable defect in this development process was the lack of a detailed prototype testing and evaluation before going to the implementation phase. This happened mainly because there were not enough resources for the project (there were not enough member in the development team to run prototype evaluating and testing in design phase) or the designers of the application assumed that they have a deep understating of the product users' needs and market and they do not need to evaluate those. At the end of development phase the whole team recognized that this neglect had a really bad impact on the result. The UI and report designer workflow in comparison

with other report designer tools could have been tested (with some easy to make and develop low fidelity prototypes like paper prototypes) with a few users to have an evaluation of the usability design decisions before spending resources on developing the application.

Although the current model of report designing implemented in this application gives users an acceptable level of freedom and a detailed control, it is complex and does not provide a smooth and efficient experience for users. The main requirement which this reporting application as a SSBI tool should support is simplicity in analyzing data and designing reports but as it was explained in the previous chapter, the implemented report design workflow, model and UI need a big improvement and redesign process to fulfil this requirement.

Now after this phase of product development, the team can have a realistic estimation about the amount of resources needed to complete this application and they can adjust the scope of the product based on their resource. The difficulties of using HTML and JavaScript as development platform must be noted, this decision must be revised or at least some ways to deal with difficulties of that should be considered. Finally, now the product team has an executable prototype they can conduct a user study and subjective assessment with real users to modify their design and improve the product for next versions based on users' feedback and result of usability evaluation.

REFERENCES

[1] Nikos Drakos, Raymond Paquet, Gartner Webinar Technology Trends You Can't Afford to Ignore, Gartner, 2009.

[2] Jennifer Rowley, Richard Hartley, in *Organizing Knowledge: An Introduction to Managing Access to Information*, Ashgate Publishing, Ltd. , 2006, pp. 5-6.

[3] http://en.wikipedia.org/wiki/DIKW_Pyramid, Wikipedia. [Online]. [Accessed 01 08 2014].

[4] Understanding and Performance, 2004. [Online]. Available: http://www.nwlink.com/~donclark/performance/understanding.html. [Accessed 1 14 2014].

[5] O. P. Rud, Business Intelligence Success Factors: Tools for Aligning Your Business in the Global Economy, John Wiley & Sons, Inc, 2009.

[6] Top 10 Strategic Technology Trends for 2012, Gartner, 2012.

[7] Top 10 Strategic Technology Trends for 2013, Gartner, 2013.

[8] The 2011 IBM Tech Trends Report, IBM, 2011.

[9] The Current State of Business Analytics: Where Do We Go From Here?, Bloomberg Businessweek Research Services, 2011.

[10] Claudia Imhoff, Colin White, Self-Service Business Intelligence: Empowering Users to Generate Insights, TDWI -The Data Warehousing Institute, 2011.

[11] Online transaction processing, Wikipedia, [Online]. Available: http://en.wikipedia.org/wiki/Online_transaction_processing. [Accessed 17 6 2014].

[12] Creating and Using Data Warehouses Overview, Microsoft, [Online]. Available: http://technet.microsoft.com/en-us/library/aa906002(v=sql.80).aspx. [Accessed 17 6 2014].

[13] OLTP vs. OLAP, datawarehouse4u, [Online]. Available: http://datawarehouse4u.info/OLTP-vs-OLAP.html. [Accessed 17 6 2014].

[14] Celina M. Olszak , Ewa Ziemba, Approach to Building and Implementing Business Intelligence Systems, *Interdisciplinary Journal of Information, Knowledge, and Management,* vol. 2, 2007.

[15] Usama Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth, From Data Mining to Knowledge Discovery in Databases, *AAAI 97,* 1996.

[16] P. N. Finlay, Introducing decision support systems, Oxford, UK Cambridge, Mass, NCC Blackwell: Blackwell Publishers, 1994.

[17] E. Turban, Decision Support and Expert Systems: Management Support Systems, Prentice Hall, 1995.

[18] D. J. Power, WHAT IS A DSS?, 1997.

[19] Business intelligence, Wikipedia, [Online]. Available: http://en.wikipedia.org/wiki/Business_intelligence. [Accessed 16 6 2014].

[20] Neil Chandler, Bill Hostmann, Nigel Rayner, Gareth Herschel, Gartner's Business Analytics Framework, Gartner, Inc., 2011.

[21] Rajiv Sabherwal, Irma Becerra-Fernandez, Business intelligence : practices, technologies, and management, Wiley, 2010.

[22] B. Evelson, Topic Overview: Business Intelligence, Forrester, 2008.

[23] B. Evelson, Want to know what Forrester's lead data analysts are thinking about BI and the data domain?, Forrester, [Online]. Available: http://blogs.forrester.com/boris_evelson/10-04-29-want_know_what_forresters_lead_data_analysts_are_thinking_about_bi_and_data_domain. [Accessed 14 6 2014].

[24] Hewett, Baecker, Card, Carey, Gasen, Mantei, Perlman, Strong and Verplank, ACM SIGCHI Curricula for Human-Computer Interaction, ACM SIGCHI, [Online]. Available: http://old.sigchi.org/cdg/cdg2.html. [Accessed 12 6 2014].

[25] E. R. Tufte, *Visual Design of the User Interface*, Armonk, N.Y: IBM Corporation, 1989.

[26] N. Elmqvist, Embodied Human-Data Interaction, *The 29th ACM International Conference on Human Factors in Computing Systems*, 2011.

[27] Hamed Haddadi, Richard Mortier,Derek McAuley, Jon Crowcroft, Human-data interaction, University of Cambridge, 2013.

[28] Richard Mortier, Hamed Haddadi, Tristan Henderson, Derek McAuley, and Jon Crowcroft, *Challenges & Opportunities in Human-Data Interaction.*

[29] Jeffrey Heer, Michael Bostock, and Vadim Og ievetsky, Tour through the Visualization Zoo, *acmqueue*, 2010.

[30] Stuart K. Card, Jock Mackinlay , Ben Shneiderman, Readings in Information Visualization: Using Vision to Think, 1999.

[31] D. A. Norman, Emotion and design: Attractive things work better, *Interactions Magazine*, 2002.

[32] C. Chen, Top 10 Unsolved Information Visualization Problems, Drexel University College of Information Science and Technology, 2005.

[33] Usability, wikipedia, [Online]. Available: http://en.wikipedia.org/wiki/Usability. [Accessed 18 8 2014].

[34] Usability - ISO 9241 definition, W3C, [Online]. Available: http://www.w3.org/2002/Talks/0104-usabilityprocess/slide3-0.html. [Accessed 18 8 2014].

[35] J. Nielsen, Usability 101: Introduction to Usability, Nielsen Norman Group, [Online]. Available: http://www.nngroup.com/articles/usability-101-introduction-to-usability/. [Accessed 18 8 2014].

[36] J. Scholtz, *Usability Evaluation*, National Institute of Standards and Technology.

[37] Usability Testing, [Online]. Available: http://www.usability.gov/how-to-and-tools/methods/usability-testing.html. [Accessed 1 9 2014].

[38] D. Kieras, *A Guide to GOMS Model Usability Evaluation using GOMSL and GLEAN4*, University of Michigan, 2006.

[39] Andrew Sears, Julie A. Jacko, The Human-Computer Interaction Handbook: Fundamentals, Evolving Technologies and Emerging Applications, in *Model-based Evaluation*, CRC Press, 2002.

[40] B. Evelson, The Forrester Wave™: Self-Service Business Intelligence Platforms, Q2 2012, Forrester Research, Inc., 2012.

[41] Methods, UsabilityNet , [Online]. Available: http://www.usabilitynet.org/tools/methods.htm. [Accessed 12 7 2014].

[42] About Gartner, Gartner, [Online]. Available: http://www.gartner.com/technology/about.jsp. [Accessed 9 7 2014].

[43] Gartner Magic Quadrant, Gartner, [Online]. Available: http://www.gartner.com/technology/research/methodologies/research_mq.jsp. [Accessed 9 7 2014].

[44] Rita L. Sallam, Joao Tapadinhas, Josh Parenteau, Daniel Yuen, Bill Hostmann, Magic Quadrant for Business Intelligence and Analytics Platforms, Gartner, 2014.

[45] B. Evelson, Who are the BI Personas?, forrester, [Online]. Available: http://blogs.forrester.com/boris_evelson/10-03-07-who_are_bi_personas. [Accessed 1 7 2014].

[46] Self-Service Business Intelligence:A Workbook, Jaspersoft Corporation, 2013.

[47] P. McFadden, What is dashboard reporting?, ExcelDashboardWidgets, [Online]. Available: http://www.exceldashboardwidgets.com/what-is-dashboard/what-is-dashboard.html. [Accessed 10 7 2014].

[48] Kim Verkooji, Marco Spruit, Mobile Business Intelligence: Key Considerations for Implementations Projects, *Journal of Computer Information Systems,* 2013.

[49] Introduction to Power BI Q&A (and cloud modeling), Microsoft, [Online]. Available: http://office.microsoft.com/en-us/office365-suite-help/introduction-to-power-bi-q-a-and-cloud-modeling-HA104167933.aspx. [Accessed 11 7 2104].

[50] Embedded Analytics: The Ultimate Value-Add for Your Applications, Information Builders, [Online]. Available: http://www.informationbuilders.com/solutions/embedded_analytics. [Accessed 11 7 2014].

[51] Visokio, [Online]. Available: http://www.visokio.com/.

[52] Visualization (computer graphics), Wikipedia, [Online]. Available: http://en.wikipedia.org/wiki/Visualization_(computer_graphics). [Accessed 20 6 2014].