Analysis of Requirements Phase as a Step for Software Engineering Automation

Abdul Sattar M. Khidhir

Technical Institute
Foundation of Technical Education, Iraq

Noor L. Shihab

College of Computer Sciences and Mathematics
University of Mosul, Mosul, Iraq

Received on: 15/04/2012 **Accepted on: 28/06/2012**

ABSTRACT

Requirements engineering is the first phase of the software engineering process. This phase is the basis of other phases, so it is a very critical phase. Incomplete, ambiguous and misunderstanding requirements lead to fail in the product. For this reason, attention must be increased in requirements engineering to obtain accurate, complete and unambiguous requirements to satisfy users' needs in their products.

In this paper, an attempt to analyze the requirement phase and produce analysis model is presented by using EA (Enterprise Architect) tool to offer a smart support for requirements engineering processes and introducing them as basis in requirements development in software engineering automation.

Keywords: Enterprise Architect, software engineering automation.

تحليل طور المتطلبات كخطوة لأتمتة هندسة البرمجيات

عبد الستار محمد خضر

المعهد التقنـي / الموصل
هيئة التعليم التقنـي، العراق

نور لؤيد شهاب

كلية علوم الحاسوب والرياضيات
جامعة الموصل، الموصل، العراق

تاريخ استلام البحث: 2012/04/15 **تاريخ قبول البحث: 2012/06/28**

الملخص

تعتبر هندسة المتطلبات الطور الأول من أطوار عمليات هندسة البرمجيات وهي الأساس للأطوار اللاحقة ولهذا السبب تكون هذه المرحلة حرجة.إن المتطلبات غير الكاملة, الغامضة و غير المفهومة تؤدي إلى فشل المنتج. ولهذا السبب زاد الاهتمام بهندسة المتطلبات للحصول على متطلبات كاملة وواضحة لتلبية حاجات المستفيدين في أنتاجاتهم.

ولقد حاولنا في هذا البحث تحليل مرحلة المتطلبات وتقديم نماذج تحليل باستخدام أداة EA (Enterprise Architect) لغرض تقديم دعم ذكي لعمليات هندسة المتطلبات وعرضها كأساس لتطوير المتطلبات لأتمتة هندسة البرمجيات.

الكلمات المفتاحية: أداة Enterprise Architect ، أتمتة هندسة البرمجيات.

1. Introduction

As software engineering plays a more important role in any types of today's systems development, more cost and effort are invested in system development and produce more successful systems.

The requirements engineering is the first process in software engineering processes. Since the 1970s, people have realized that getting the right requirements is a prerequisite for successful software development [7].

What is the requirement? The simple definition of the requirement is what the customer requires the system to do.

According to IEEE software engineering glossary, it defines a requirement as

(1) A condition or capability needed by a user to solve a problem or achieve an objective;

(2) A condition or capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed documents.

(3) A documented representation of a condition or capability as in (1) or (2).[7]

Requirements should only state what a system is supposed to do rather than to describe the way to do it[7]. It is the input to the requirement engineering process that is defined as the process of determining what is to be produced in a software system, is an important aspect of any software project, and is a general term used to encompass all the activities related to requirements.[3]

In this paper, more details about requirement engineering process, requirement types, analysis of these types and determination if the process of these requirements can be automated are provided.

2. Background

2.1 Requirements engineering and requirement process[7]

There are several definitions for the term requirement engineering. The different definitions reflect the different aspects of requirements engineering and show that requirements engineering has been expanded in all aspects including the scope, the life cycle, the activities in the process, and the roles involved in the process.

The general requirements engineering definition is:

"A systematic process of developing requirements through an iterative, co-operative process of analyzing the problem, documenting the resulting observations in a variety of representation formats and checking the accuracy of the understanding gained. It has been invented to cover all of the activities involved in discovering, documenting, and maintaining a set of requirements for a computer-based system".

Requirements engineering has emerged in order to better understand how the software interacts with the other parts of the system, what needs to be further developed in a systematical way, and helps to identify and structure requirements.

Finally, a definition of requirements engineering process is given :

The requirements engineering process is "a structured set of activities which are followed to derive, validate and maintain a systems requirements document".

The requirements engineering sub processes (as shown in Figure 1) are

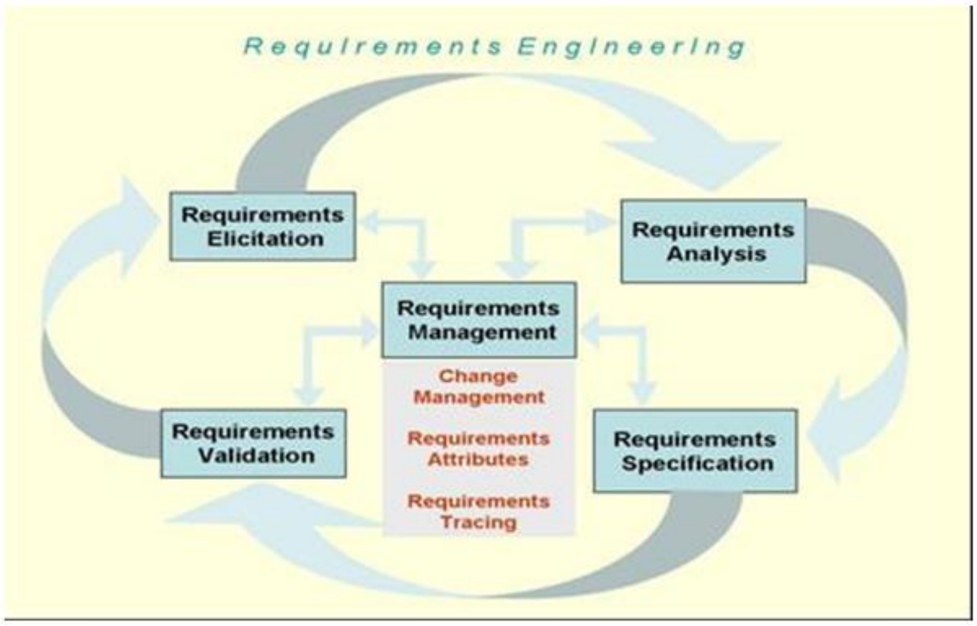

Figure 1: Requirements Engineering Processes[9]

a) Requirements Elicitation

Requirements Elicition is concerned with where software requirements come from and how the software engineer can collect them. It is the first stage in building an understanding of the problem the software is required to solve. It is fundamentally a human activity, and is where the stakeholders are identified and relationships established between the development team and the customer. It is variously termed "requirements capture," "requirements discovery," and "requirements acquisition."[1]

There are three main categories of participants: the developer, the user and the customer. In addition, major requirements elicitation can involve other people such as lawyers, government standards organizations and so on.[3]

b) Requirements Analysis

Encompasses those tasks that go into determining the needs or conditions to meet for a new or altered product, taking account of the possibly conflicting requirements of the various stakeholders, such as beneficiaries or users[10] the process of analyzing requirements concerned with Detect and resolve conflicts between requirements, Discover the bounds of the software and how it must interact with its environment and Elaborate system requirements to derive software requirements. Therefore, it is critical to the success of a development project.[1]

c) Requirements Specification

For most engineering professions, the term "specification" refers to the assignment of numerical values or limits to a product's design goals. The understanding obtained by problem analysis forms the basis of requirements specification, in which the focus is on clearly specifying the requirements in a document called Software Requirements Specification(SRS). Issues such as representation, specification languages, and tools are addressed during this activity.[1][6] Requirements might be documented in various forms, such as natural-language documents, use cases, user stories, or process specifications.[10]

d) Requirements Validation

The requirements documents may be subject to validation and verification procedures. The requirements may be validated to ensure that the software engineer has understood the requirements, and it is also important to verify that a requirements document conforms to company standards, and that it is understandable, consistent, and complete. The aim is to pick up any problem before resources are committed to addressing the requirements. Requirements validation is concerned with the process of examining the requirements document to ensure that it defines the right software (that is, the software that the users expect).[1]

The requirement process is the sequence of activities that need to be performed in the requirements phase and that culminate in producing a high-quality document containing the Software Requirements Specification (SRS). The requirements process typically consists of three basic tasks

a) problem or requirement analysis.
b) requirements specification.
c) requirements validation.

The overall requirement process is shown in Figure 2 .[6]

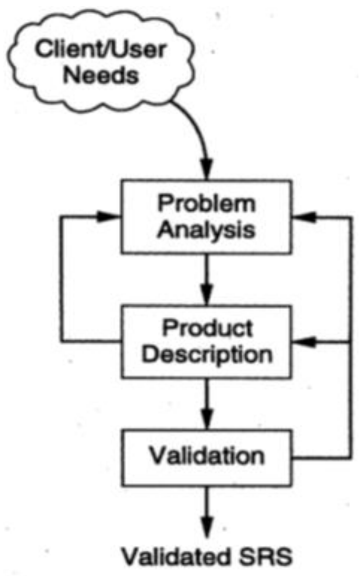

Figure 2: The Requirement Process.[6]

2.2 Requirements Types

Requirements types are logical groupings of requirements by common functions, features and attributes. There are more than one classification of requirements, They are

i. <u>State/Mode Requirements:</u>

State the required states and/or modes of the item, or the required transition between one state and another state, one mode and another mode, made in one state to mode in another state, or the response required of the system as a direct consequence of a transition having occurred. A "state" is a condition of something required or permitted. A "mode" in this context is a related group of functionality for a purpose, i.e. "mode of operation".[8]

ii. <u>Functional Requirements</u>

Functional requirements state what the system is required to do. It specifies the expected outputs of the system should be produced from the given inputs. They describe the relationship between the input and output of the system. For each functional requirement, a detailed description of all the data inputs and their source, the units of measure, and the range of valid inputs must be specified.[8]

iii. <u>Performance Requirements</u>

Performance requirements state how well the system is to do what it is to do. That is, performance is an attribute of function. It specifies the performance constraints on the software system.[8]

iv. <u>External Interface Requirements</u>

External Interface requirements state the required characteristics at a point or region of connection of the system to the outside world (e.g. location, geometry, inputs and outputs by name and specification, allocation of signals to pins etc).[8]

v. <u>Environmental Requirements</u>

Environmental requirements limit the effect that external environment (natural or induced) is to have on the system, and/or the effect that the system is to have on the external enveloping environment.[8]

vi. Resource Requirements

Resource requirements limit the usage or consumption by the system of an externally provided resources.[8] Resource requirements such as (hardware, software, documentation,….etc).

vii. Physical Requirements

Physical requirements state the required physical characteristics (properties of matter) of the system as a whole (e.g. mass, dimension, volume, centre of gravity, surface coefficient of friction, density, etc).[8]

viii. "Other Quality" Requirements

Other quality requirements state any other required quality, that is not one of the above types, nor is a design requirement.[8]They are: Correctness, Completeness ,Consistency ,Clarity, Non-ambiguity ,Traceability ,Testability ,Singularity , Feasibility ,Balance and Freedom from product/process mix.

ix. Design Requirements

Design requirements direct the design (internals of the system), by inclusion (build it this way), or exclusion (don't build it this way).[8]

x. Data requirements

Define the specific data items or data structures that must be included as part of the software product. For example, a payroll system would have requirements for current and year-to-date payroll data.[5]

xi. User requirements

They define what the software has to do in order for the look at the functionality of the software product from the user's perspective. They define what the software has to do in order for the users to accomplish their objectives. [5]

xii. Business requirements

Define the business problems to be solved or the business opportunities to be addressed by the software product. In general, the business requirements define why the software product is being developed. Business requirements are typically stated in terms of the objectives of the customer or organization requesting the development of the software.[5]

xiii. Behavioral Requirements

Behavioral requirements explain what has to be done by identifying the necessary behavior of a system.[10]

xiv. Architectural Requirements

Explain what has to be done by identifying the necessary system architecture of a system. A system architecture or systems architecture is the conceptual model that defines the structure, behavior, and more views of a system. It makes use of elements of both software and hardware and is used to enable design of such a composite system. [10]

xv. Structural Requirements

Explain what has to be done by identifying the necessary structure of a system. Structure is a fundamental, tangible or intangible notion referring to the recognition, observation, nature, and permanence of patterns and relationships of entities. The description of structure implicitly offers an account of what a system is made of: a configuration of items, a collection of inter-related components or services. A structure may be a hierarchy (a cascade of one-to-many relationships), a network featuring many-to-many links, or a lattice featuring connections between components that are neighbors in space. Data structure is one type of structure.[10]

xvi. Derived Requirements

Requirements that are implied or transformed from higher-level requirement. For example, a requirement for long range or high speed may result in a design requirement for low weight.[3]

xvii. Allocated Requirements

A requirement that is established by dividing or otherwise allocating a high-level requirement into multiple lower-level requirements. Example: A 100-pound item that consists of two subsystems might result in weight requirements of 70 pounds and 30 pounds for the two lower-level items.[3]

Some writers(like Linda Westfall) classify requirements into levels and each level contains one or more types of requirements for example : one of them classifies requirements into six types within three levels according to table1[5]:

Table1: Requirements Levels

Levels	Requirements types
Business	Business
User	User
Product	Functional
	Nonfunctional
	External interface
	Data

2.3 Prior work in Requirements Engineering

The Users' Involvement in The Requirements Engineering Process was studied by Daniela Elena Herlea[3] said: If developers and end-users are in different organizations or different cities, meetings can be costly, inconvenient and infrequent. This leads to problems of communication, which can greatly impact the quality of the elicited requirements.

The system called TeamRooms integrates the rich applications and interfaces found in the existing real-time groupware applications, providing a persistent work space suitable for synchronous and asynchronous collaboration. TeamRooms may be used as a method of distance collaboration between the knowledge engineer, the developer who elicits the information, the domain expert and the knowledgeable end-user.[3]

The support for the requirements engineering process was studied by Qian Zhang discussed the outcome of his research is the Generalized Attribute-centered Requirements Engineering Process (GAREP) and its tool. GAREP is a generalized version of the Requirements Assistant for Telecommunication Services (RATS) methodology. The GAREP tool is a distributed system that provides intelligent support from the early stages of requirements development. The architecture of the GAREP tool is based on a web-based software framework, and can facilitate the implementation of good practices in this process.[7]

The Requirements Engineering for Software Product Lines was studied by Chethana Kuloor and Armin Eberlein[2] said : A product line is a set or group of products that has a majority of features in common and varies only in certain specific features.

In order to obtain high-quality products along with higher productivity, it is required to carefully analyze, model, specify and manage system requirements. In that paper, requirements engineering practices used in the existing product line approaches are analyzed and the applicability of several existing requirements engineering techniques to product line development is assessed.[2] The requirements engineering process produces a set of complete, right, unambiguous and accurate requirements that you can store them in order to retrieve them when you need it in related project.

So in The Building Requirements Repository Using Requirements Transformation Techniques to Support Requirements Reuse study by Sami M. Homod and David C. Rine they applied the reuse techniques at the requirements engineering phase of the software development life cycle will help to get complete and accurate requirements with less effort and time. Sami and David developed an approach for building a conceptual requirements repository to support the requirements reuse process. The completed and corrected requirements specification is the target of requirements engineering phase. One of the most important applications of software components reuse is the requirements reuse to facilitate and enhance the process of requirements engineering.[9]

3. Model for Automation

The result of analysis for any project is an" analysis model". In this work the requirement phase was analyzed and an analysis model was produced using EA (Enterprise Architect) tool, in order to determine which kind of requirements can be performed automatically to help the software engineering. The initial step for requirements analysis is shown in the Figure 3.The values of tags in this figure are to be filled by the software engineer or analyst with the help of the customer. Performance requirement, physical requirement, behavioral requirement, functional requirement, and other quality requirement are part of quality layer(quality level) whereas others are not part of quality layer even though quality is inherent in them.
The decision for requirement automation is presented in this work.

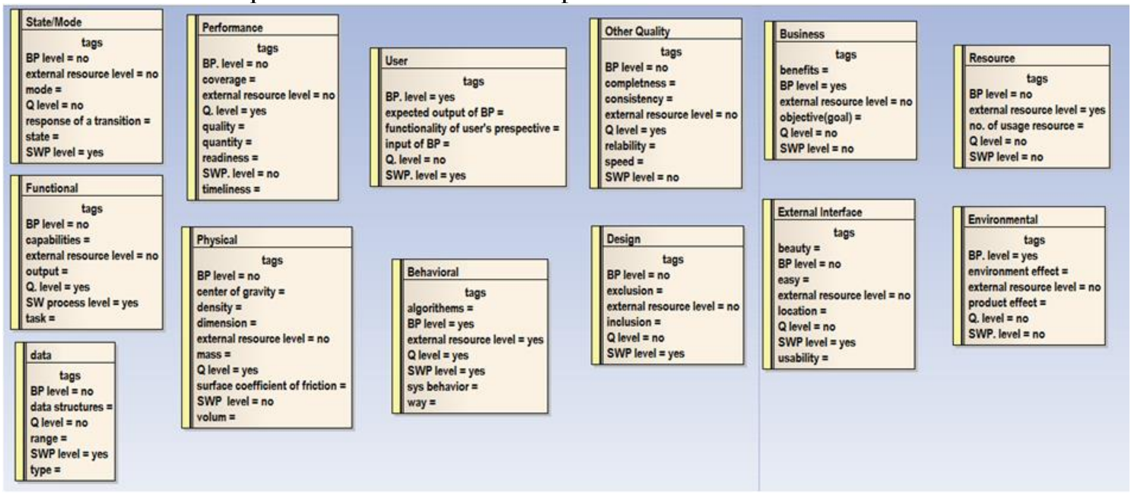

Figure 3: Requirement Diagram for Requirements Type

In the meeting that discusses the business requirement model Figure 4, it is shown clearly that the meeting activity cannot be automated although it can use computer aided software by using web connection. Thus, the customer writes his requirements (if they leave far from each other) and in the other side the analyst checks it or they are meeting through video conferencing. This is very helpful if the members (customer, analyst/software engineer) are not in the same place or they are busy.

A program whose result is a good template of System Requirements Specification (SRS) can be written to specify customer requirements in it. The analyst can separate manually system objectives and benefits from customer requirements.

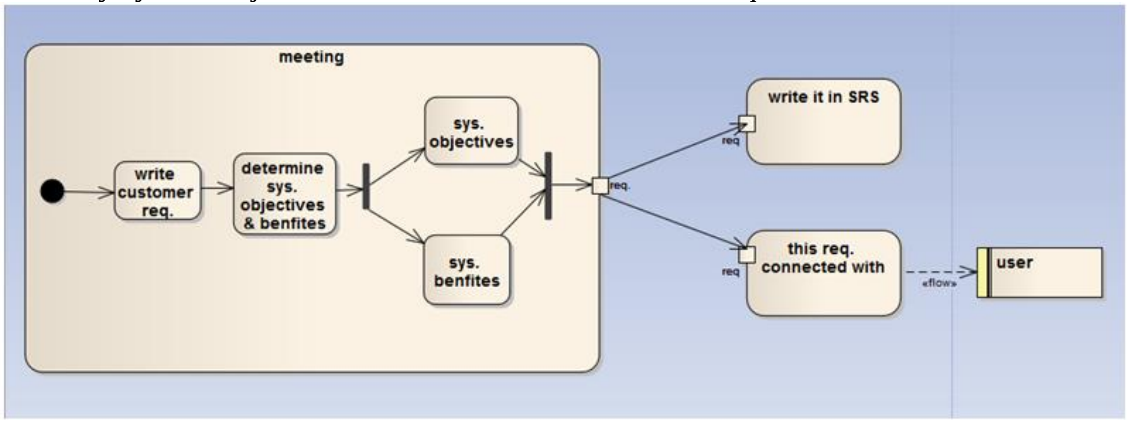

Figure 4: Activity Diagram for Business Requirements

After determining user requirements , data requirements and the input /output of the project by the customer, the developer finds the way or the program to reach user satisfaction and business requirements. Sometimes the developer adds or changes user input, with the help of user, in order to produce the output that the latter wants as shown in Figure 5.

Processing this input (function requirements) can be made in an automated way, but we can't make a program that fits all projects because these projects differ from each other. A data- base can be built to store the input to be used in other similar projects.

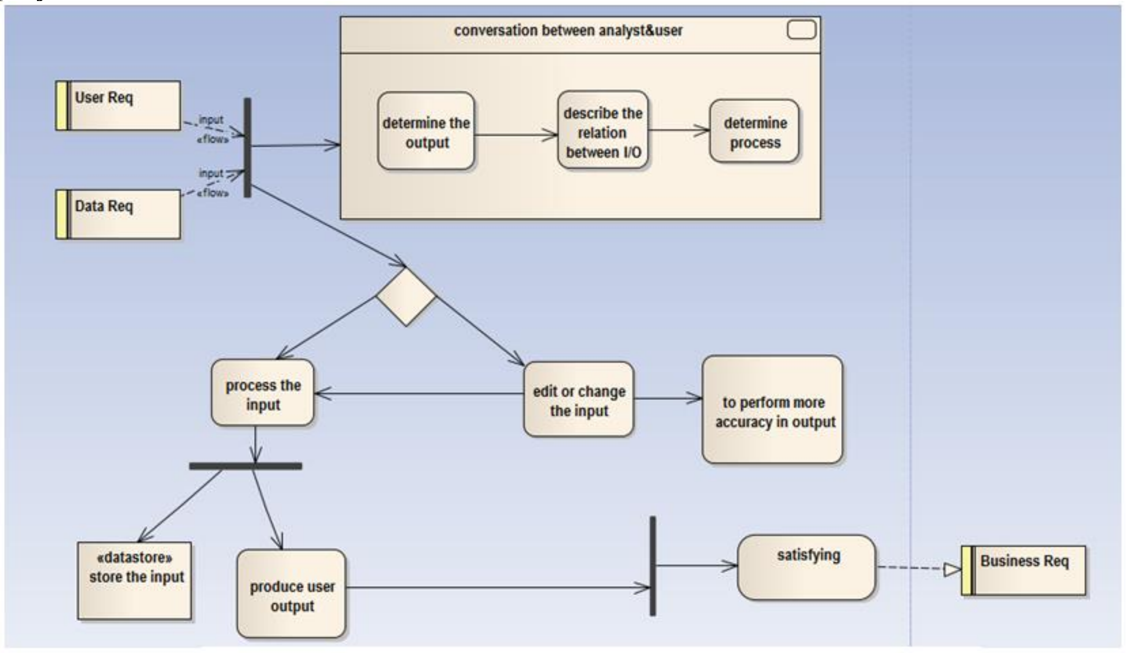

Figure 5: Activity Diagram for Functional Requirements

In Figure 6 the customer conversation about the needed resources in the project is analyzed. In some cases these resources are complete, in others the developer adds some other needed resources in his work to achieve the customer's needs The resources are humans, software and tools. This model can't be performed automatically, but it can be computerized. The analyst, after specifying the resources, submits a report including all the needed resources together with their qualities, quantities and cost.

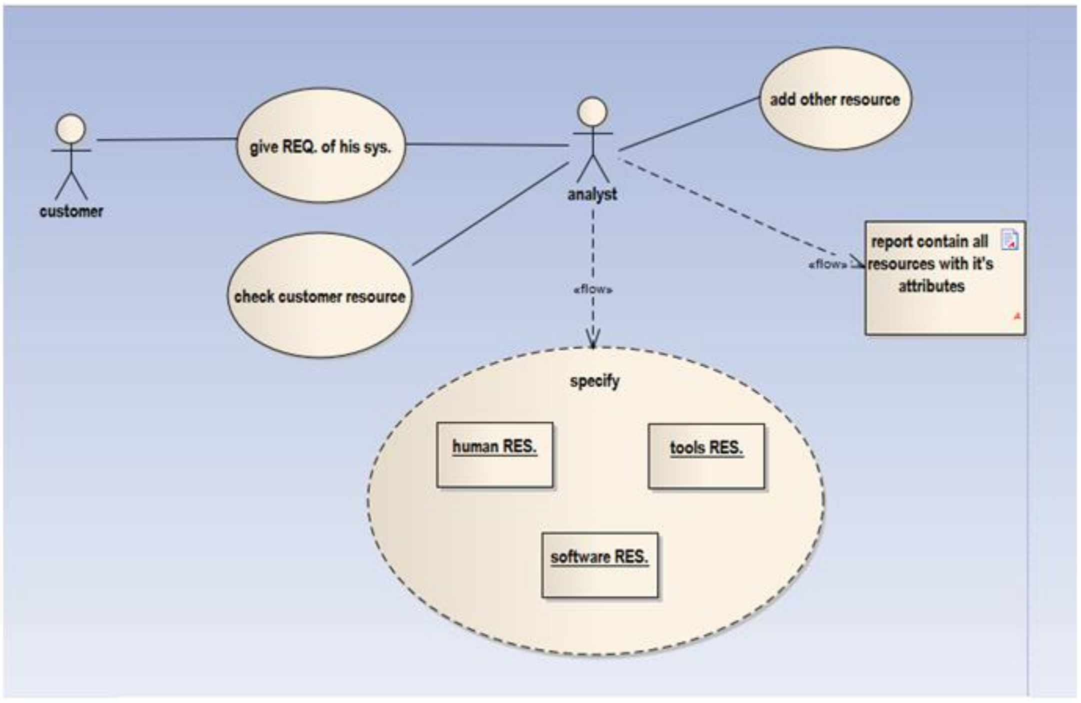

Figure 6: Use Case Diagram for Resource Requirements

The behavioral requirements determine the way that the system(product) follows to reach its objective. The behavior of any type of requirement can be specified if these requirements have known data and algorithms .In this case, those types of requirements can be designed and implemented as shown in Figure 7.

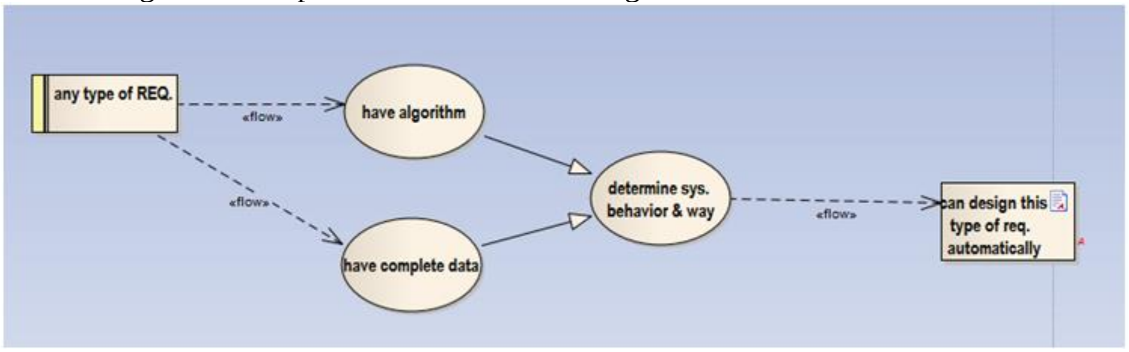
any type of REQ.
«flow»
have algorithm
«flow»
have complete data
determine sys. behavior & way
«flow»
can design this type of req. automatically

Figure 7: Use Case Diagram for Behavioral Requirements

Data requirements result from either the customer identification or from the developer specifications. They are used in functional requirements as the process needs after determining their types, ranges and data structure as shown in Figure 8 . This data can be saved in a data base to use it in some other future projects.

The requirement that doesn't have data is a non-functional requirement. The process that checks if the requirement has data or not can be made in an automated way.

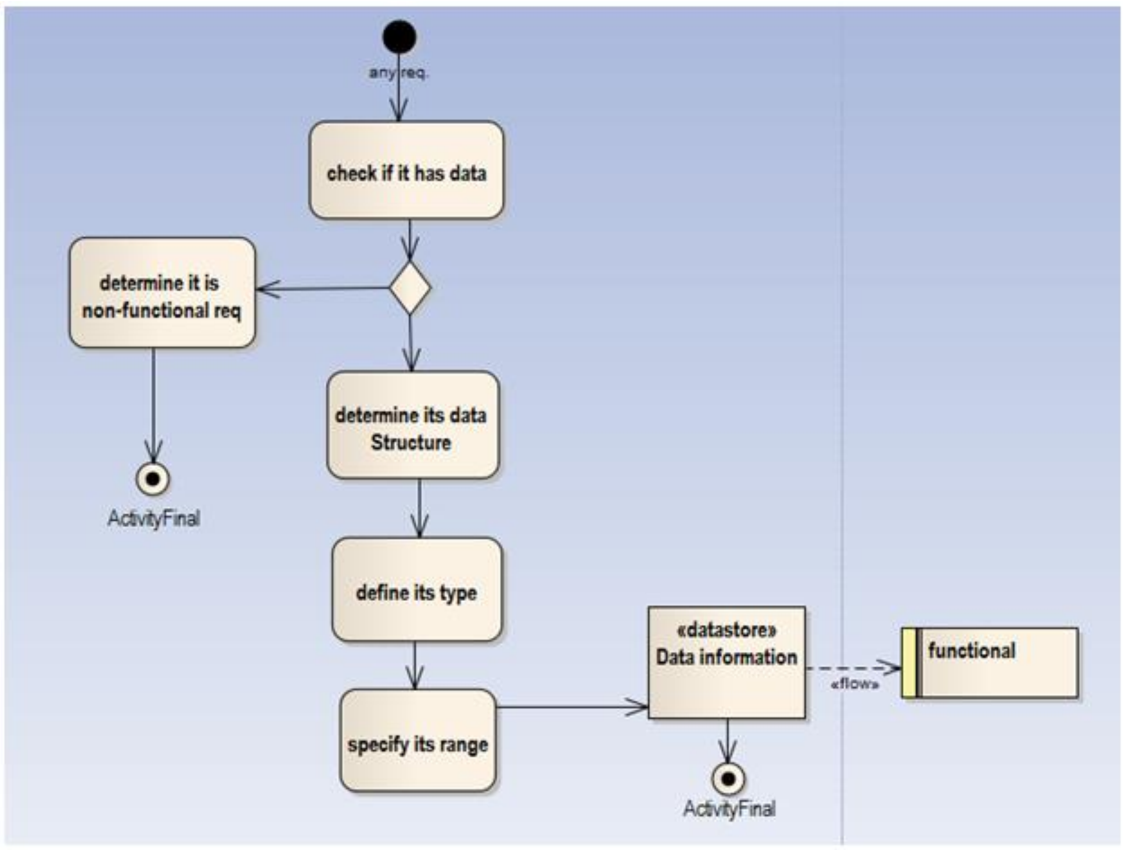

Figure 8: Activity Diagram for Data Requirements

Conclusion

The analysis model is very helpful to make a complete and accurate design compared with textual document. For this reason, software engineering diagrams are implemented(using EA tool). In this research determining the requirements that can be performed automatically without the software engineer help is tried. For example, functional requirements can be made in an automated way by writing a program processing user and data requirements to produce user outputs or business requirements.

In the behavioral requirements a check program should be built that checks XML document of a model if that model has data and algorithm. If such a model has data and algorithm it can be converted into design model automatically and its behavior(way) can be found.

The business requirements activities cannot be made automatically, but can use computer aids. For example, the meeting activity can be computerized and applied a program whose result can be a good template of SRS.

The analyst, with the help of the computer can produce a report containing all the resources needed in his projects. The result of the resource requirements can be computerized to provide a guide report to help the software engineer to supply the resources.

REFERENCES

[1] Alain Abran, and James W. Moore, "Chapter2: Software Requirements", Guide to the software engineering body of knowledge, (SWEBOK), 2004.

[2] Chethana Kuloor, and Armin Eberlein, "Requirements Engineering for Software Product Lines", The University of Calgary, Alberta, Canada, 2002.

[3] Daniela Elena Herlea, "Users' Involvement In The Requirements Engineering Process", University of Calgary, Alberta, Canada, 1996.

[4] Department of Defense of U.S.A, Department of Defense Handbook, "System Requirements Document Guidance", MIL-HDBK-520(USAF) 5 March 2010.

[5] Linda Westfall, "Software Requirements Engineering: What, Why, Who, When, and How", The Westfall Team, 2006.

[6] Pankaj Jalote, "A Concise Introduction to Software Engineering", Springer-Verlag London Limited, 2008.

[7] Qian Zhang, M.Sc. thesis, "Support for the Requirements Engineering Process, The University of Calgary, Alberta, Canada, (2003).

[8] Robert Halligan, "Types of Requirements", http://www.ppi-int.com, accessed at 2nd Feb. 2012 .

[9] Sami M. Homod and David C. Rine, "Building requirements Repository Using Requirements Transformation Techniques to Support Requirements Reuse", Proceedings of the IIIS Conference on Information Systems Development, 1999.

[10] http://en.wikipedia.org/wiki/requirments_analysis, accessed at 5th Oct. 2011.

Appendix

In this appendix, explanations of tags of Figure 3are presented. Each requirement in this figure has tags (attributes).

tags	Means
BP level	Business Process level that some requirements belong to. This requirements deals with BP (e.g. user requirements).The value yes: the requirement included in this level. The value no: the requirement did not include in this level.
SWP level	Software Process level that some requirements belong to (e.g. design requirements). The value yes: the requirement included in this level. The value no: the requirement did not include in this level.
Q level	Quality level that some requirements belong to (e.g. quality requirements). The value yes: the requirement included in this level. The value no: the requirement did not include in this level.
External Resource level	Level that some requirements belong to (e.g. resource requirements). The value yes: the requirement included in this level. The value no: the requirement did not include in this level.
Mode	Is a related group of functionality for a purpose, i.e. "mode of operation".
State	Is a condition of something required or permitted.
Response of transition	The response required of the system as a direct consequence of a transition having occurred between one state and another state, one mode and another mode, and made in one state to mode in another state.
Capabilities	Define the capabilities of the software product.
Output	The output that the customer wants from the products functions.
Task	Identifying the necessary task that must be accomplished to satisfy functional requirements.
Data structure	Defines the structure of the product's data needed in the product.
Range	Specifies the range of the data in the product.
type	Determines the type of the data in the product.
Coverage	A term for measuring performance requirements.
Quality	A term for measuring performance requirements.
Quantity	A term for measuring performance requirements.
Readiness	A term for measuring performance requirements.
Timeliness	A term for measuring performance requirements.
Center of gravity	One of the physical characteristics (properties of matter) of the system.
Density	One of the physical characteristics (properties of matter) of the system.
Dimension	One of the physical characteristics (properties of matter) of

	the system.
Completeness	The degree of the completeness of the activities in the product.
Consistency	The range of the consistency between product activities .
Reliability	Identifies the degree of the reliability of the activities in the product.
Speed	Determines the speed of the activities execution in the product.
Exclusion	Identifies the parts excluded in the design(don't build it this way).
Inclusion	Identifies the parts included in the design(build it this way).
Benefits	Define the benefits of the product(system).
Objective(goal)	Defines the goal of the product(system).
Beauty	Defines the degree of beauty of interfaces in the product.
Easy	Determines the degree of easiness in the use of the product interface .
Location	Determines the location of the external interface whether it is with user or with other systems.
Usability	Determines if the product is easy and helpful in use.
Volume	One of the physical characteristics (properties of matter) of the system.
Surface coefficient of friction	One of the physical characteristics (properties of matter) of the system.
Mass	One of the physical characteristics (properties of matter) of the system.
Expected output of BP	Determines the expected output of Business Process defined by the stakeholders.
Input of BP	Determines the input of Business Process defined by the stakeholders.
Functionality of user's perspective	Identifies the functionality of the product from user's perspective.
Way	Determines the way to implement product activities.
No. of usage resource	Number of the resources used in the product.
Environment effect	Specifies if there is an effect from the environment on product.
Product effect	Specifies if there is an effect from the product on the environment or other systems.
Algorithms	Determine the algorithm of the requirement (if it has).
System behavior	Defines the behavior of the system.