STEP IT ACADEMY

SOFTWARE PROJECT MANAGEMENT

Lesson 3

REQUIREMENTS, TOOLS USED WITHIN THE PROJECT

CONTENTS

1. WHAT IS REQUIREMENT?

In the software development industry, there are several definitions of the term "requirement"; however, we will rely on the following:

> ***Requirement*** *is a condition or capability needed by a user to solve a problem or achieve an objective.*
>
> © IEEE Standard Glossary of Software Engineering Terminology (1990)

On the Internet, you can find many illustrations demonstrating the importance of working with requirements. One of the diagrams shows the following statistics:

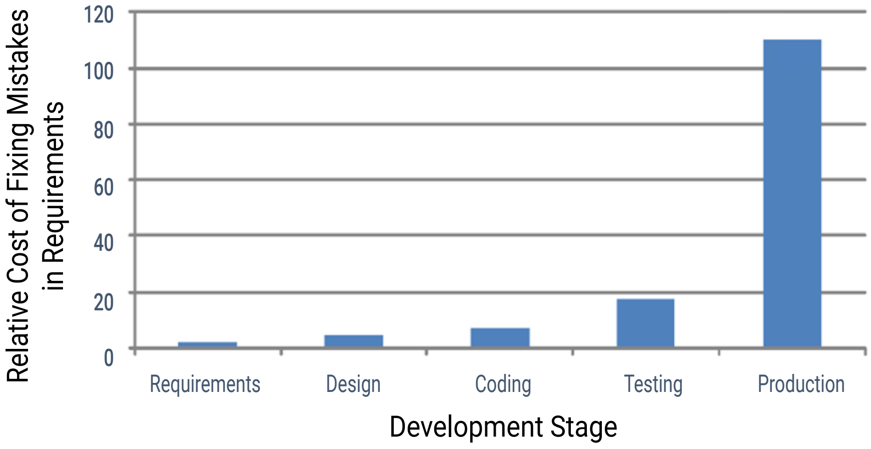


Figure 1

This diagram shows dependence of the cost of correcting mistakes in the requirements depending on the development stage. The cost of fixing mistakes during the production stage of the created software product may exceed the cost of all that was previously done in the project.

You can find other data demonstrating the importance of requirements management for a software product:

- Completeness of requirements collected by the method of discussion with the customer is 25%-50%, reaching a peak of 70%-80%;
- Completeness of identifying such requirement type as quality attributes is 50%-60% on average, i. e. without special attention to this type of requirements, the analyst can identify about half of all quality attributes that are important to the customer of the project.

In IT projects there is a very interesting effect: almost always a wall of misunderstanding arises between the customer and the team. The customer speaks the business language, and the team speaks the “Martian IT language”. A business analyst is called upon to solve the problem of misunderstanding.

Business analyst is a separate profession in the IT industry, which requires a lot of effort and time to master. Business analysts have a body of knowledge that incorporates information on business analysis, **BABOK** (Business Analysis Body of Knowledge).

BABOK summarizes the experience of leading business analysts around the world and describes 30 key business analysis problems and 50 common techniques used to solve them. BABOK is used to train specialists for the certification exams of the International Institute of Business Analysis (**IIBA**).

The objectives of the business analyst in the implementation of the project may include:

- **Collection (extraction) of requirements**. At this stage, the analyst needs to collect the most detailed requirements for the product of the project (for example, the information system);
- **Requirements analysis**. It is necessary to check the collected requirements for completeness and feasibility. Some requirements may be too complex to implement or impossible to implement using the selected technologies;
- **Identification of problems and contradictions**. At this stage, the analyst must make sure that the requirements do not contradict each other, that each formulated requirement has no contradictions.

  An example of contradictions in the requirements is described here:

  https://www.ijedr.org/papers/IJEDR1402214.pdf.
- **Develop solutions for the problems found**. For each formulated contradiction in the requirements, it is necessary to find a solution that will be adequate from the point of view of the cost of its implementation and timing.
- **Research**. In projects, there are situations where an analyst needs additional research to clarify the requirements. For example, a client complains that the number of leads keeps decreasing. The analyst needs to analyze the site and develop ways to improve this.
- **Forecasting**. It is important for some IT projects that the analyst can tell the client about trends in their field of

activity and forecasts for the development of this business. For example, an analyst can tell a client about new business models of their competitors.

More details about business models can be found in the book ***Business Model Generation: A Handbook for Visionaries, Game Changers, and Challengers.***

To solve all these problems, a good business analyst should have the following qualities:

- Analytical thinking;
- System thinking;
- Excellent communication skills;
- Attention to details and good time management skills;
- Creativity;
- Aptitude for learning.

According to the statistics of **The Standish Group** (the world leader in software research), one of the key reasons for the failure of IT projects is incomplete requirements for the product:

Project Impaired Factors	**% of Responses**
Incomplete Requirements	13. 1
Lack of User Involvement	12. 8
Lack of Resources	10. 6
Unrealistic Expectations	9. 9
Lack of Executive Support	9. 3

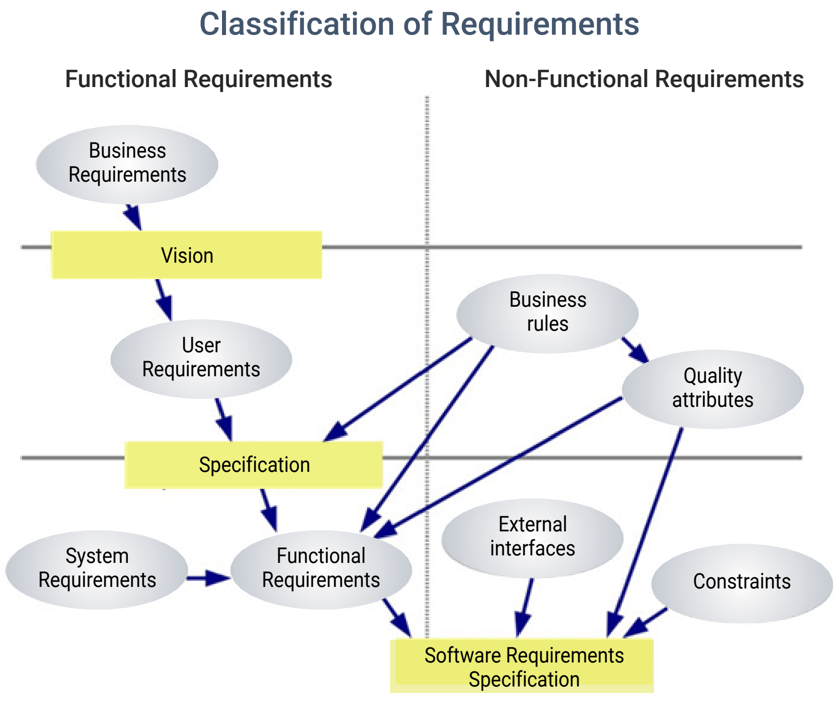
Project Impaired Factors	**% of Responses**
Changing Requirements & Specifications	8. 7
Lack of Planning	8. 1
Didn't Need It Any Longer	7. 5
Lack of IT Management	6. 2
Technology Illiteracy	4. 3
Other	9. 9

© The Standish Group. Chaos report, 2015

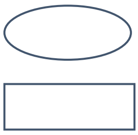
Since work with requirements becomes one of the key factors for the success of an IT project, in the following sections we will consider the process of working with requirements and classification of requirements.

2. CLASSIFICATION OF REQUIREMENTS TO THE SOFTWARE PRODUCT

In order to not miss important requirements to the software product, business analyst should use classification of software requirements. One way to classify requirements was proposed by **K. Wiegers**:



Legend:



– type of information for requirements;

– ways to store information (documents, diagrams, database).

Figure 2

Business requirements answer the question: why do you need a software product?

User requirements describe the goals and tasks that users will be able to solve using the software product

Functional requirements describe what the information system should do to enable users to accomplish their tasks and achieve their goals.

System requirements describe the high-level characteristics of the information system, including:

- requirements for work under certain operating systems;
- hardware requirements;
- requirements for support lines after the implementation of the software product.

Business rules describe corporate policies, standards, procedures, and so on.

Quality attributes describe the information system in terms of characteristics important to users.

Examples of quality attributes:

- Ease of use (*usability*);
- Scalability;
- Integration with other company information systems.

Constraints are conditions that limit the choice of possible solutions for the implementation of individual requirements or their sets.

External interfaces describe the interaction with other systems and the operating environment. These include requirements for API of the product or system, as well as requirements for APIs of other systems that are being integrated.

Knowing about the existing requirement types for the software product, the business analyst is unlikely to overlook some requirement type. This means that they will at least ask the head of the IT project whether it is necessary to collect the requirements of this type and, if yes, they will collect and analyze the requirements of this type.

It is noticed that business analysts most often focus on the collection and analysis of business requirements, user and functional requirements, while forgetting about other types of requirements.

Qualities of Good Requirements

How can you understand how good the requirements for the software product are? To do this, you need to understand what qualities the requirements must have. In the literature there are such qualities of good requirements:

- Complete (i.e. sufficient);
- Correct (from the authors' point of view);
- Feasible (in the given conditions);
- Necessary (for some reason and for someone);
- Prioritized (by importance);
- Unambiguous ;
- Verifiable (try it yourself).

On the one hand, the qualities of good requirements are obvious, but in practice, it is very difficult to develop such requirements.

3. WHAT IS REQUIREMENTS DEVELOPMENT?

The process of working with requirements consists of three activities:

1. Plan to work with requirements.
2. Requirements development.
3. Management of requirements change.

In turn, **requirements development** consists of:

- Collection (extraction) of requirements;
- Analysis;
- Documenting requirements;
- Verification.

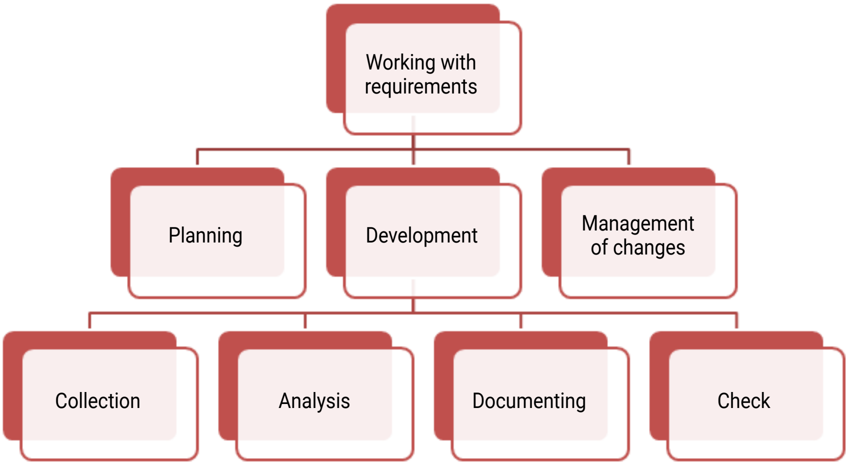


Figure 3

Planning requirements for the Software product is to:

- think about what tools (approaches) to use to extract requirements;
- coordinate these tools with the project manager;
- determine which specific groups of users for this product you need to extract requirements from and coordinate it with the project manager.

Actions of requirements development can be described as follows:

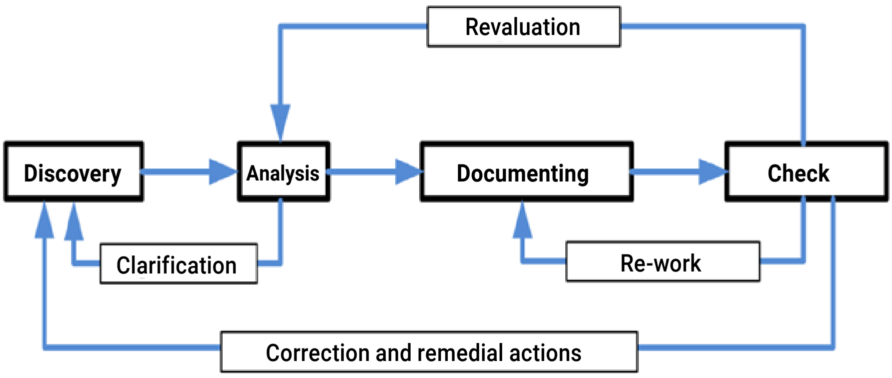


Figure 4

The analyst begins their work with discovering the requirements.

At the stage of discovery, the business analyst should extract the requirements from the stakeholders of the project (*approaches to extracting the requirements will be written below*).

Each requirement must be analyzed for its feasibility, evaluated for the scope of work for implementation and cost of implementation, and discussed with the customer of the project whether this requirement should be implemented considering its cost. During the analysis, it may be necessary to clarify something in the requirements.

After this, the requirement and approach to its implementation should be placed in the SRS.

At the final stage, the business analyst must make sure that the developers implemented the requirement correctly. If implementation differs from description in the document, then it is possible to remake the document according to the requirements. Also, according to the results of the implementation, the project customer can make some refinements to the requirements, and this will lead to get back to the analysis stage or even to the discovery stage.

Actions for requirements management may include:

- Keeping SRS version updated;
- Analysis of proposed changes to requirements and evaluation of the impact of each change on the scope of project activities;
- Development of tasks for implementation of approved changes to requirements;
- Making sure that tasks from the project plan meet the requirements;
- Tracking statuses of the requirements.

4. RISKS IN REQUIREMENTS DEVELOPMENT

1. **Lack of stakeholders involvement (including users)**

Product developers do not always understand the importance of interacting with product users about requirements, and users often do not know what exactly they expect from a software product. Aside from: "I want it to be user-friendly and fast," they rarely extract requirements on their own.

Lack of stakeholders' involvement may lead to omission of important requirements, which will lead to the need to spend unplanned time for their discussion and implementation. All these factors will lead to a sharp decrease in the likelihood for the project to meet the deadline and budget.

2. **Growing user demands**

Users and the customer of the product tend to generate ideas about new requirements as the project progresses. Some of these requirements are valuable and could be fully implemented. However, if the project has a fixed cost, the profitability of this project will be reduced for the developer team as new requirements are added.

The project owner and the team must agree on how they will manage changes in requirements (*see below*) at the start of the project. Otherwise, the team will prevent adding new requirements into the project.

3. Ambiguity of requirements

There are two symptoms of the ambiguity of requirements. The first is that the user can interpret the same requirement in two or more ways.

Another symptom is that people who study the requirement have different idea of how this will work in the product. After the requirements are implemented, they often say: "It works not as it supposed to" or "Not as we expected."

To reduce this risk, the project manager and business analyst should ask several users to express their views on the understanding of the requirement. The second way to reduce risk is to develop a testing option for the requirement and a prototype.

4. Gild the lily

The meaning of this risk is that developers add functions to the product that are not in the SRS. They believe that new features will appeal to users. However, it's often the case when users request a feature or an interface element which has no particular value for the product.

To reduce the gilding effect, test each user requirement for relation to business requirement.

5. Minimum specification

Project customers often do not understand the importance of detailed requirements and are not ready to pay for it. This tempts the team to create a simplified version of SRS or cut it to the list of business requirements. This increases the probability that the scope of work, timing, and budget of the project will not be calculated correctly.

6. **Skipping user groups**

   If the business analyst misses a group of users for your product, some requirements will not be included in the project work plan which will lead to errors in estimating the timing and budget for the project. To reduce this risk, you need to identify all classes of users and make sure that representatives of these classes will be heard by the business analyst.

7. **Careless planning**

   It all starts with the question: “I want to implement this thing for our product. When can you do it?” An imprudent answer to this question may lead to an obligation that will sink your project. It’s better to answer like this: “I'm 80% sure that it can be implemented. However, I need to consult the team and evaluate the implementation costs and timing for this thing.”

8. **Slow requirements discussion with users**

   Users are often busy with day-to-day work, they don’t have time to participate in questionnaires, they do not want to read SRS, etc. Minimize this risk by calculating slack time for tasks on the coordination of requirements, and add sanctions to the contract imposed for exceeding deadline on coordination of requirements.

9. **Slow users’ feedback on the implementation of requirements**

   After the team implements some requirements, they want to receive feedback from users, but users do not want to participate in meetings to review the implemented requirements or reschedule these meetings.

5. C REQUIREMENTS (CUSTOMER REQUIREMENTS) AND D REQUIREMENTS (DEVELOPER REQUIREMENTS)

There is one more approach to requirements classification (described in the book "**Software Engineering. An Object-Oriented Perspective**" by **Eric J. Braude**, 2004) which divides requirements into 2 groups:

1. **Customer requirements (C requirements)** are written in a language understandable to the customer of the project, they reflect their expectations and needs.
2. **Developer requirements (D requirements)** are written in a language understandable to the development team, and structured in the usual manner.

If we use the above requirements classification by **K. Wiegers**, we can conditionally separate the C requirements and D requirements as follows:

Classification of C and D requirements	Requirements Classification by Wiegers
C requirements	Business requirements User requirements Business rules Quality attributes

Classification of C and D requirements	Requirements Classification by Wiegers
D requirements	Functional requirements System requirements External interfaces Constraints

Both classes of requirements appear in the SRS document, which structure was given earlier.

If you use the IEEE SRS template, the C requirements should be included in the “Introduction" and “Overall Description” sections, and the “Specific Requirements” section includes the D requirements.

6. APPROACHES TO EXTRACTION OF C REQUIREMENTS

The most common approaches for extracting requirements are shown in the figure below:

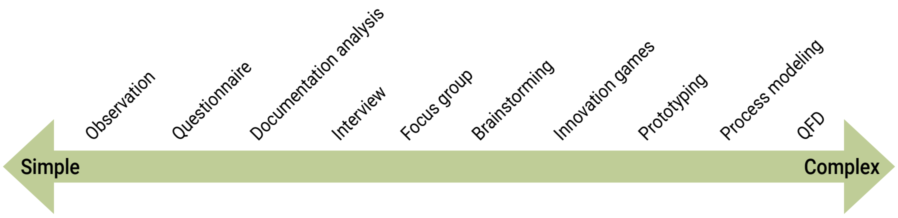


Figure 5

Approaches to the extraction of requirements are arranged on the scale by their complexity: simple approaches are on the left, complexity of approaches grows as you go from left to right. Please note that such ranking is a subjective opinion of several business analysts.

- **Observation** involves obtaining information about how users work with the product.

Observation methods may vary: you can gather statistics about how often certain functions are used, or you can team up with the business analyst and observe together.

For example, a business analyst can go to the customer's office to observe how employees interact with the software. Based on the results of observations, they lay down requirements.

- **Questionnaires** ask questions and get written responses from the project stakeholders.

This approach is used when the participant only needs to choose one or more answers or if the interview is very expensive.

- **Documentation analysis** involves the study of enterprise standards, procedures and regulations related to the rules and sequence of business processes that the team is going to automate.
- **Interview** is the most common approach to extracting C requirements that suggests a personal meeting between the business analyst and the stakeholder.

The analyst prepares questions for the interview in advance, sets place and time for the meeting. During the interview they ask questions, observe non-verbal signals of the stakeholder, ask clarifying questions, and write down the answers. At the end of the interview, the analyst gives the interviewee an opportunity to read the written result of the interview and make corrections.

- **Focus groups** is a tool that came from marketing. It allows you to gather several representatives of users and other stakeholders, and have a meeting; as a result you can get a list of requirements for the product.
- **Brainstorming**. This approach has long been known, and is used to gather a wide range of ideas about requirements.
- **Innovation games** (*business games*) is an approach popular in the United States developed by **L. Hohmann**. The idea is to involve the project stakeholders in gathering of requirements and turn this into a fun but profitable game.

- **Prototyping** is the most common method of gathering requirements in **Agile**. The development team creates a prototype of several functions of the software product and presents it to future users. The prototype allows you to get feedback and clarify the requirements.
- **Process modeling** involves graphical models of user activity.

Interviews, questionnaires, and focus groups are often used to collect information before modeling. There are a lot of process modeling notations, but the most popular ones are **IDEF0**, **eEPC**, **BPMN 2.0**, **UML** (especially **use cases**), **DFD** (data flow diagrams).

- **SQFD (Software quality function deployment)** is an approach that originated in Japan as **QFD**, which was later used in software development.

QFD uses a complex analysis of the cause and effect matrix between the customer's wishes and technical characteristics of the product.

SQFD can also be used for requirements analysis.

7. APPROACHES TO THE ANALYSIS OF C REQUIREMENTS

After the C requirements are extracted, the business analyst should analyze them for completeness, consistency, and feasibility. The business analyst can use the following tools for analysis:

- Backward requirements analysis;
- Analysis of analogical systems;
- TIPS (theory of inventive problem solving);
- Root Conflict Analysis Plus (RCA+);
- Value-Conflict Mapping+;
- Quick prototyping.

Approaches to Extraction of D Requirements

To implement C requirements, the development team must lay down detailed D requirements.

See mapping of SRS sections according to IEEE 830 into D requirements in the figure 6.

To develop D requirements for the product, developers can use the following models:

- use-case diagrams of UML;
- Data Flow Diagrams (DFD);
- UML statechart diagrams.

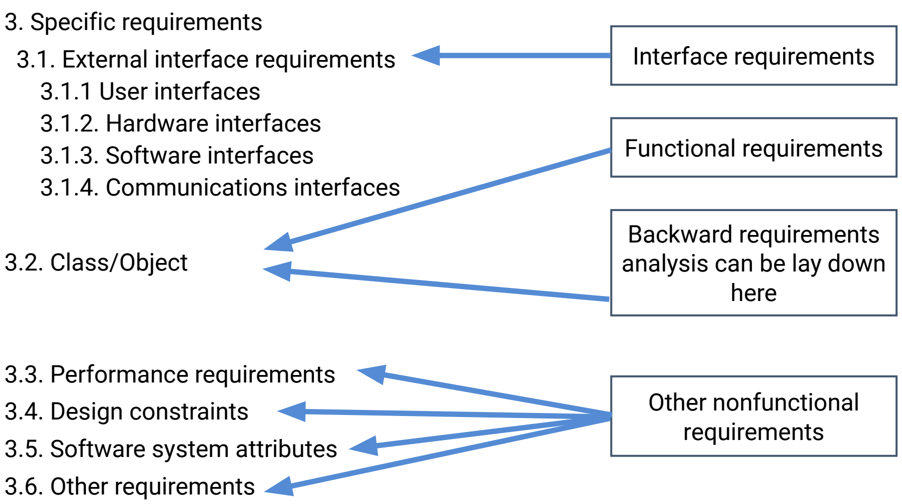

3. Specific requirements
3.1. External interface requirements
3.1.1 User interfaces
3.1.2. Hardware interfaces
3.1.3. Software interfaces
3.1.4. Communications interfaces
3.2. Class/Object
3.3. Performance requirements
3.4. Design constraints
3.5. Software system attributes
3.6. Other requirements
Interface requirements
Functional requirements
Backward requirements analysis can be lay down here
Other nonfunctional requirements

Figure 6

Rules for using diagrams include the following steps:

1. If the requirement is simple and affects other requirements, then diagrams are not used, and the requirement is given verbally in a suitable SRS section.
2. If the requirement is the interaction between the user and the application, then it is expressed through a use-case diagram.
3. If the requirement affects processing elements, each of which receives and outputs data, data flow diagrams (DFD) are used.
4. If the requirement affects states of the program, statechart diagrams are built.

Example of Turning a C Requirement into a D Requirement

A business analyst formulated **C requirement** as follows:

To record customer's requirement for shipping, the forwarder should create a shipping request.

Document details are given in the table below.

Table 1. Shipping request details

No.	Name	Short Description
1	Document number	
2	Date	
3	Customer	
4	Forwarder	
5	Sender	
6	Recipient	
7	Notification	Text: "Notify party of arrival"
8	Instructions	Text: "The Customer instructs, and the forwarder undertakes the obligation to deliver the cargo"
9	Cargo name	
10	Country of origin of the cargo	
11	HS code	
12	Marking	
13	Cargo weight (total, nett)	
	Size	

No.	Name	Short Description
	Cost	
14	Number of cargo items	
15	Type (size) of packaging	
16	Ship from	Point of departure or cargo transfer to the forwarder
17	Ship to	Point of destination
18	Shipping deadline	Date of cargo delivery
19	Mode of carriage	Cargo delivery methods: type(s) of transportation, single wagon load, shipload, aggregate shipment, containerized shipment, etc.
20	Shipping instruction	Cargo properties, special instructions, other cargo information
21	Documents required	
22	Cargo insurance	Details about cargo: insured, to be insured, and so on
23.1	Responsibilities of the parties	Text: "Responsibilities of the parties in addition to those listed in the contract"
23.2	Responsibilities of the forwarder	
24	Responsibilities of the Customer	
24.1	Liabilities of the parties	Text: "Liabilities of the parties in addition to those listed in the contract"
24.2	Liabilities of the forwarder	
25	Liabilities of the Customer	
25.1	Payment policy	

No.	Name	Short Description
25.2	Cost of services of forwarder	
26	Payment arrangements and due dates	
27	Expire date	
28	Other details (special notes)	
29	Forwarder's name	
30	Forwarder's signature and seal	
31	Customer name	
32	Customer's signature and seal	

The below diagram displays state of a request in the system:

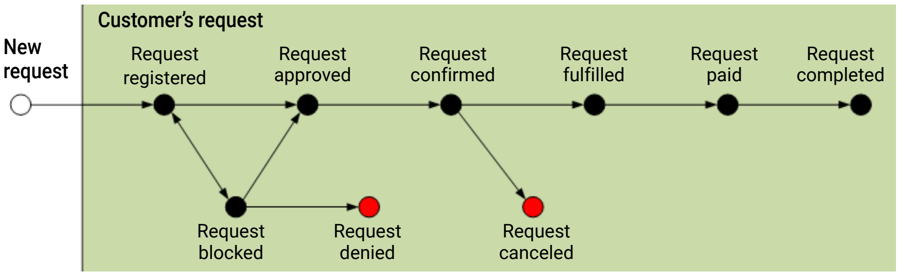


Figure 7

The request gets into the system during registration. The below table lists description of the request states.

Table 2. States of the customer's request

No.	State	Description
1	Request registered	Request data are entered in the system. Number is assigned to it.
2	Request approved	The request was verified and can be fulfilled. The fact that the request was approved doesn't mean that it will be fulfilled. See item 5.
3	Request blocked	The request was not approved due to non-conformity.
4	Request denied	The request can be denied due to one of the following reasons: 1. Customer's receivable has exceeded its limit, and conditions of the request fulfillment cannot be changed appropriately. 2. Customer's receivable will exceed its limit, and conditions of the request fulfillment cannot be changed appropriately.
5	Request approved	Approval of request is sent to the customer as Forwarder's Instructions.
6	Request canceled	Changes made to the request from this point forward are documented, saved, and coordinated with the Customer again.
7	Request fulfilled	Cancelation means that the request fulfillment is suspended by the company or the Customer. If financial liabilities appear (e.g. penalty), they must be satisfied.
8	Request paid	Mutual settlements between the Customer and the company related to the request are carried out.
9	Request completed	Request is closed for editing by users.

In order to turn **C requirement** to **D requirement**, business analysts decided to create use-cases with requests in the system.

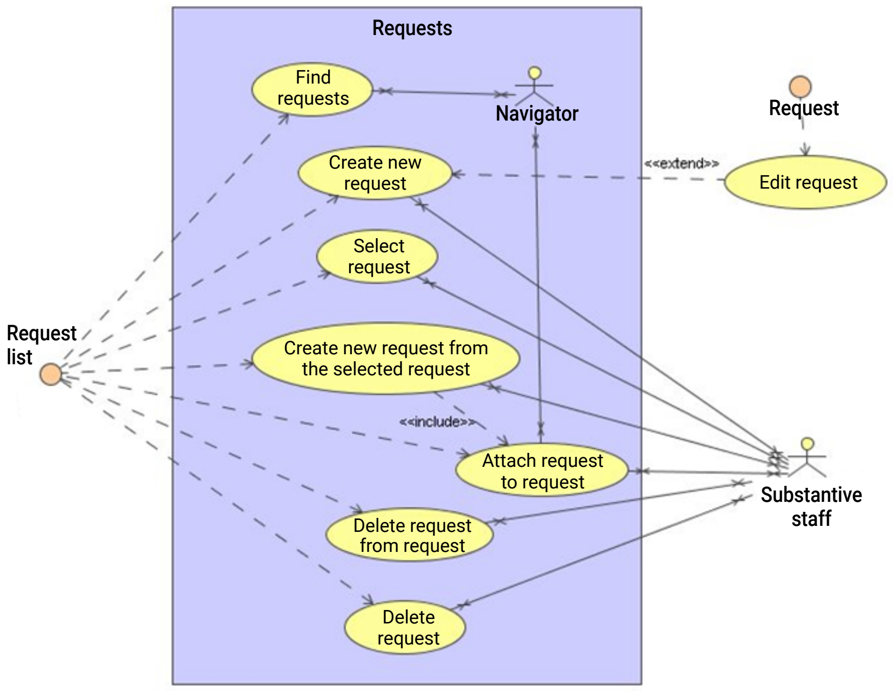


Figure 8

To create the **Request** object and description of its interaction with other objects in the system, business analysts described the data structure of the request.

Table 3. Data structure of the Request object

General parameters

No.	Document	Description
1	Document number	Document number in the system
2	Date	Date and time when the request was registered in the system (assigned automatically)
3	Ref No.	No. in the Customer's request
4	Contract	
5	Contact name	
6	Currency	

Cargo

Cargo description. The **Cargo** subordinate directory is called.

Transport

No.	Tag	Short Description
1	Type of transport	
2	Machine	The **Machine** directory is called
3	Equipment	
4	Comment	

Services

No.	Tag	Short Description
1	Service	
2	Included in wage	

No.	Tag	Short Description
3	Amount	
4	Amount, EUR	

Checkpoints

The **Checkpoints** directory is called.

Request parameters

No.	Tag	Short Description
1	Organization	
2	Unit	
3	Office	
4	Main curator	
5	Destination curator	
6	Request group	
7	Transaction	
8	Destination	
9	Transaction program	
10	Type of shipping	
11	Loading date (plan)	

Request Conditions

The request is executed on the basis of the conditions specified in the relevant contract. The Forwarder can edit the terms of the contract within the specified limits, as well as add other conditions.

The request conditions are a list where a separate list item is a separate condition. The data structure describing the request condition is presented in the following table.

Data Structure of the Request Condition

No.	Tag	Short Description
1	Term of payment	
2	Term of penalty	
3	Additional conditions	
4	Comment	

Term of Payment

Only one term of payment can be assigned to the request. Data structure describing the term of payment is given in the following table.

Data Structure of the Term of Payment in the Request

No.	Tag	Short Description
1	Condition text	Is not edited in the request. Changed automatically upon editing of separate parameters of the term of payment in the request
2	Type of payment	Is set in the contract. Is not edited in the request. Possible values: due upon presentation of shipping documents, for the period
The Prepaying Section		
3	Prepaying, %	Is set in the contract. Is edited in the request within the allowed range
4	Type of bill	Is set in the contract. Is not edited in the request. Possible values: original, copy (fax), copy (email)

No.	Tag	Short Description
5	CP of the pay period	Is set in the contract. Is not edited in the request
6	Payment period (before, after)	Is set in the contract. Is not edited in the request. Possible values: before, after
7	Payment time period	Is set in the contract. Is edited in the request within the allowed range
The Payment Section		
8	Type of bill	Is set in the contract. Is not edited in the request. Possible values: original, copy (fax), copy (email)
9	CP of the pay period	Is set in the contract. Is not edited in the request.
10	Payment period (before, after)	Is set in the contract. Is not edited in the request. Possible values: before, after
11	Payment time period	Is set in the contract. Is edited in the request within the allowed range

Business analysts included this **D requirement** in the Specific Requirements section of SRS.

8. DOCUMENTING REQUIREMENTS

To document requirements, you typically use the SRS document templates; their structure was given in Lesson 2.

To automate requirements documenting and work with requirements documents, you can use special software products, such as **Confluence**, **Zoho Wiki**, or free **Google Docs**.

9. CHANGE REQUIREMENTS MANAGEMENT

Let's consider how management of the requirements change is implemented when using the **waterfall model** of the project lifecycle, and when working with the iteration model used in **Agile**.

If the Customer understands that not all product requirements make it into the first version of the concept, and the project timing and budget are fixed, the team must have a list of prioritized requirements for the product.

Priorities for the requirements help the Customer quickly make decisions on what requirements to the product to withdraw if the deadlines approved in the contract are coming to an end and the budget is not enough to implement all requirements. But the Customer should make sure that if some requirements are refused, the product will still be able to solve the identified problems of future consumers.

With this approach, the management of changes in requirements is reduced to constant ranking of the remaining requirements. This approach is used in **Scrum**.

Consider an option of fixed requirements and flexible time and budget that can change as the project goes.

In this case, the requirements for the products are worked up before the product implementation starts, as a rule, and it is assumed that the requirements to the products will not change. However, as requirements are implemented and the prototype of the product appears, the initial requirements may not seem relevant to the customer of the project, or new requirements may appear.

In this case, the project manager should be ready to negotiate with the customer about the need to make changes in the SRS, about the change in cost and timing of the project. If changes in requirements are approved, they should initiate the signing of an addendum to the contract.

To work with changes, a special document **"Change Request"** is introduced and the decision-making algorithm looks as follows:

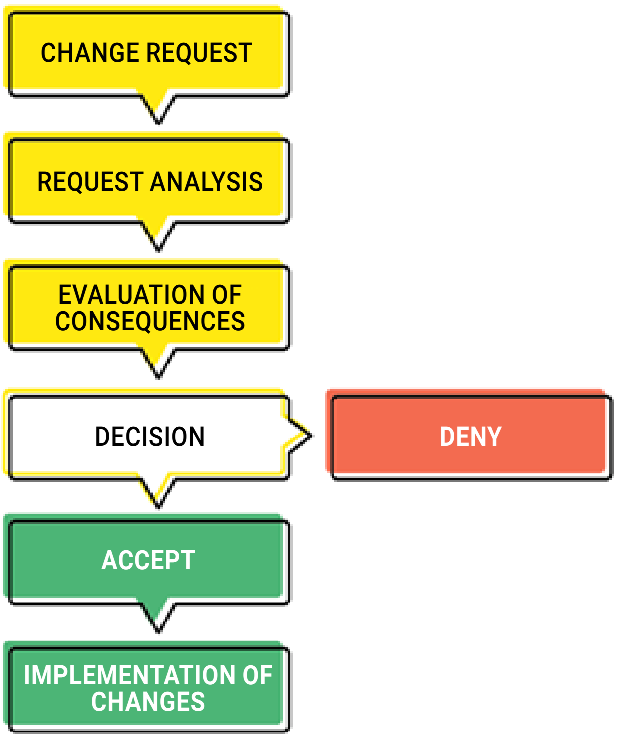


Figure 9

In order to make a decision on change, the project manager, the business analyst, and the customer must go through many steps (Steps 2 and 3 are when the business analyst is involved in the project, the customer should be involved in Step 4). As a result, it takes a lot of time and energy to process each change request. But without it the scope of the project will increase without adjusting the time and budget, which will lead to a breakdown of the originally approved deadlines and budget.

Takeaway:

- Good work with the project requirements greatly increases the likelihood of project success.
- Skipping requirements discovered in the later stages of the IT project is likely to lead to serious costs for its implementation.
- The main person responsible for working with the requirements in the IT project is business analyst.
- All requirements for the software product can be divided into two groups: C requirements and D requirements, and it is worth adding the K. Wiegers' classification.
- Business analysts need to master a large number of methods and tools to extract requirements.
- Requirements must be documented in SRS which the customer of the IT project must approve.
- As the requirements are implemented, and the first prototype of the future product appears, the customer may want to make changes to the list of requirements. There are

two options for managing changes in requirements, and one of them must be approved by the customer before the project starts.

HOME TASK

1. Find a person who is not satisfied with the website of their company. Interview this person and write what problems the current version of the website has.
2. Analyze these problems and describe business requirements to improve this website.
3. Find several users of this site and interview them; after this create 10-15 user requirements.
4. Find an experienced web programmer and send him/her your list of user requirements so he/she could evaluate the effort.
5. Upon evaluation of the effort, the programmer may have questions on the nature of the requirements or on the way they are to be implemented. Answer his/her questions.
6. List tasks on implementation and test of the requirements, evaluate effort for each task.

List of tasks for requirements implementation::

Requirement	Tasks on the requirement implementation	Effort, man-hour	Executor

Lesson 3

REQUIREMENTS, TOOLS USED WITHIN THE PROJECT