PhD, Trunov O.M., PhD student, O.Volkova

Mykolaiv Petro Mohyla State Humanities University in consortium with “Kiev-Mohyla Academy”

REQUIREMENT PHASE IN SOFTWARE DEVELOPMENT PROCESS FOR MEDICAL DIAGNOSTIC SYSTEM

The first stage of information technologies (IT) **product**[1] development process is **requirement**[2] agreement with customer in compliance with qualifying **standards**[3] [1,2]. This paper focuses on design of diagnostic system, which is under construction, includes a **requirement statement phase**[4] and based on a Rapid Application Development Technology approach to engineering process.

Using powerful computer resources during sophisticated medical diagnostic process, analysis of patient’s electrocardiogram and definition of the correct diagnose allows processing of the digital bioelectrical heart’s potential. The following computer diagnostic systems are presented on modern IT market [3,4]: S-5 Datex (Finland): “Cardiacap”; “Cardiacare” (USA), “ARGUS PB-1000” (Switzerland); Life Scope Bedside Monitors BSM-2301 “Nihon Kohden” (Japan).

Above-listed automated medical diagnostic systems are too complicated and don’t provide complete **reliability**[5]. Causes of unreliability of presented products are the following: absence of synergy between software developer’s experience and technical requirements of doctors, that make the process of development and application medical software more complicated; incorrect definitions of product requirements or omission the software requirement phase in development process.

Phase of definition of all the requirements concepts [5] to developed medical diagnoses system for analysis of electrocardiography information is absolutely necessary to proceed to the next phase of engineering process. Necessity to define product requirements prior to starting development follows from: development team needs requirements to obtain what is actually needed from the very beginning, not rebuilding the product again and again many times during the development process; testers need product requirements to start tests development as early as possible; set of **measurable**[6] quantities or **metrics**[7] allows to manage the progress during the development process; set of measurable requirements enables more easy (or enables at all) formal acceptance of the product by customer.

The classifications of Product Requirements from books [6] and differ from requirements which are classified by authors as demonstrated by diagram 1.

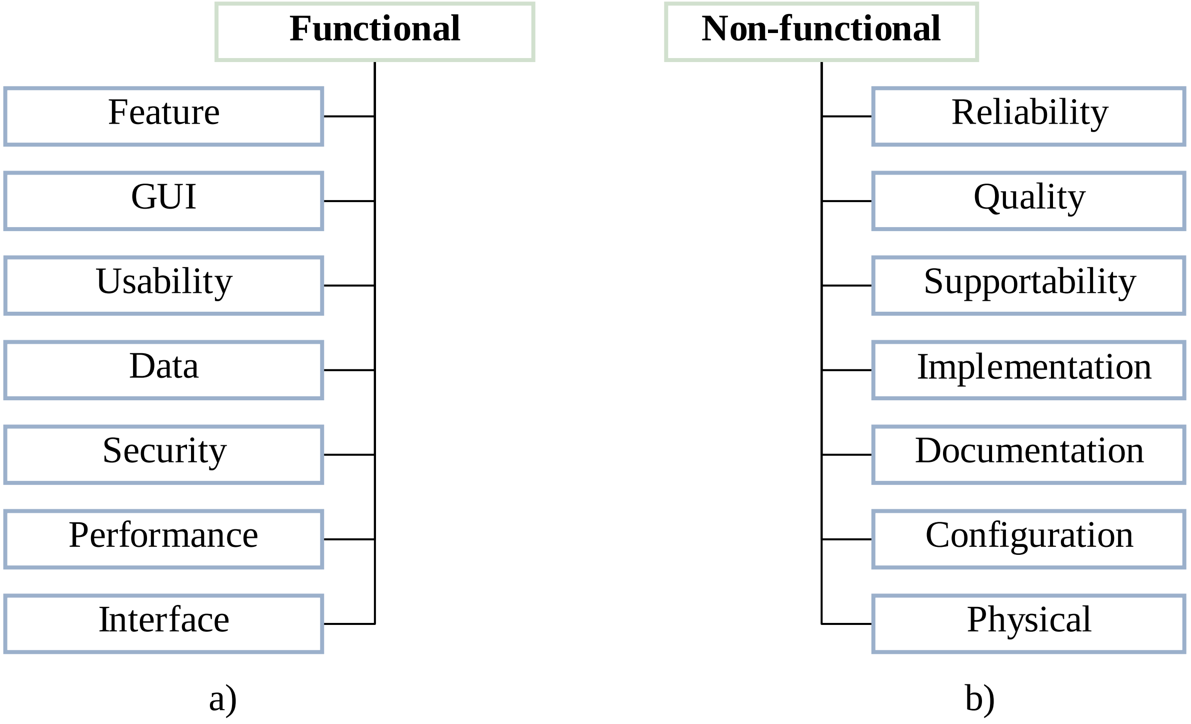

Diagram1. Product Requirements Types

Product Requirements are the representation of capability that must be met by the system or system components [7] to enable user to solve a problem/achieve an objective. PR describe what should be implemented but don't describe how it should be implemented.

Functional requirements are the system's capabilities to perform functions and services that enable user to solve a problem/achieve an objective. Among functional requirements we can pick out several sub-types as can be seen from Diagram1a and which are described below in details. *Functional (Feature) requirements* define functions and services which system should perform on the system or user level, as well as what system should not perform. *Graphic User Interface (GUI) requirements* define presence, appearance, functionality and behavior of application's GUI elements. *Usability requirements* define all human factor-concerned requirements like ergonomic, clearness, simplicity and consistency of application User Interface (UI), easiness and accessibility of UI elements. Data requirements define input and output data for the system, format of the data, what data should be saved, accuracy of calculations etc. *Security requirements* define access permissions to the system functionality and stored data, security levels and security maintaining, data backup and restore which can be performed by the user. *Performance requirements* define performance parameters for the system such as: scalability; synchronization of parameters for application functioning at any time; any limitations for simultaneous actions/resources usage by application. *Interface requirements* define interaction of the system with other systems or environments: inputs, received from these systems and outputs, directed to these systems, as well as data formats, media required for such interaction.

Non-functional requirements define conditions, restrictions and limitations which system should conform, and standards to which the system should correspond. Among non-functional requirements we can pick out several sub-types as can be seen from Diagram1b and which are described below in details. *Configuration requirements* describe what hardware/software configurations should system work with/on, as well as differences in functionality depending on configuration. *Reliability requirements* define parameters like average/maximum frequency of failures, mean time between failures, and failures distribution by severity, maximum downtime, data backup/recovery mechanisms which are implemented – algorithms, media. *Documentation requirements* define all kinds of documentation which should or should not be present for the system, ways to publish/print these documents, to who they are intended. *Implementation requirements* define any constraints as well as industry, **quality**[9] or other standards which the system should conform to: implementation languages, coding conventions etc. *Supportability requirements* define how product is supported, what are the ways of getting licenses, technical support for users, fixing problems in the released product, delivering patches and new versions to user, ways to upgrade/patch the system. *Physical requirements define* the functionality of equipment in physical conditions, restrictions concerning temperature, humidity, other environmental limitations, places to install equipment.

Each requirement should have priority assigned to it. Primary purpose of this is to prioritize requirements implementation by development team in case of lack of resources/time. An example of 3-level prioritization may be the next: P1 (most important requirements) – implementation is absolutely necessary in the current version of the product; P2 (very desirable requirements) – implementation is extremely desirable but not necessary in the current version of the product; P3 (all other requirements).

All requirements should be added to PR document as soon as they received from customer or developer such as: Vision Document (VD), Requirements Definition Document (RDD) and Software Requirements Specification (SRS). VD is destined to collect, analyze, and define high-level needs and features of the system and prepared by customer in natural language and contains all the common (high-level requirements). RDD contains all the requirements which came directly from customer and are already reviewed by PM and approved to implement. SRS contains actually the same information as previous one, but it is intended for use by system architects and developers, testers, so it utilizes language and notations used in software development. All Product Requirements documents should necessarily be kept under version control.

Concussions

Results of definition, classification and setting the priority of product requirements to computer diagnostic system allow creating SRS document and show advantages of application recommended PR classification of medical system because of easy product’s improvement. Besides SRS document comes as the input data to the next stages of product development process – design process and qualification testing, that in one's turn, directly connected with checking of Product Requirement, because the repeated verification of software accordance to specified requirements defines product quality.

Glossary

[1]**Product:** The output of a process; the work product. Products are defined by a statement of requirements; they are produced by one or more people working in a process. [2]**Requirement:** A formal statement of: 1) an attribute to be possessed by the product or a function to be performed by the product; 2) the performance standard for the attribute or function; or 3) the measuring process to be used in verifying that the standard has been met. [3]**Standards:** The measure used to evaluate products and identify nonconformance. [4]**Statement of Requirements:** The exhaustive list of requirements that define a product. [5]**Reliability:** The probability of failure-free operation for a specified period. [6]**Measurement:** The act or process of measuring. A figure, extent, or amount obtained by measuring. [7]**Metric:** A measure of the extent or degree to which a product possesses and exhibits a certain quality, property, or attribute. [8]**Quality:** A product is a quality product if it is defect free. To the producer a product is a quality product if it meets or conforms to the statement of requirements that defines the product.

Literature

1. Bailey J. J., Benson A. S., Garson A. et al. Recommendation for standardization and specification in automated electrocardiography bandwidth and digital signal processing: A report for health professionals by an ad hos writing group of the committee on electrocardiography and cardiac electrophysiology of the Council ors Clinical Cardiology. – American Heart Association // Circulation / - 1990. - Vol. 81. - P. 1-730.
2. International Organization of Standardization http://www.iso.ch/ – ISO 13485:2003 Medical devices – Quality management systems – Requirements for regulatory purposes; ISO 9126:2001, Software engineering – Product quality.
3. Meersman de R., Faroudia N, Juris P., Higgins J., Gentile A. A computerized respiratory sinus arrhythmia. program for the non-invasive assessment of parasympathetic activity // Comput. Biol. Med. – 1990. – №90. – P.75-94.
4. Uijtdehaahe S.H. A BASIC program for the peak-to-valley estimation of respiratory sinus arrhythmia // Int. J. of Bio- Medical Computing. – 1994. – №35. – P.169-192.
5. Medical Electrical Equipment. P 3: Particular Requirement for the Essential Performance of Recording and Analyzing electrocardiographs. — Geneva, 1996.
6. Heninger K. L. Specifying Software Requirements for Complex Systems: New Techniques and Their Application, IEEE Transactions on Software Engineering, vol. SE-5, № 1, January 1980, 2-13.
7. Karl E. Wiegers, Software Requirements. Second Edition. Microsoft Press (ISBN 0-7356-1879-8), 2003.