Вісник Тернопільського національного технічного університету
https://doi.org/10.33108/visnyk_tntu
Scientific Journal of the Ternopil National Technical University
2020, № 2 (98) https://doi.org/10.33108/visnyk_tntu2020.02
ISSN 2522-4433. Web: ***visnyk.tntu.edu.ua***

UDC 004.91

SOFTWARE DEVELOPMENT DOCUMENTING – DOCUMENTATION TYPES AND STANDARDS

Anna Synko; Andriy Peleshchyshyn

Lviv Polytechnic National University, Lviv, Ukraine

***Summary**. Due to the rapid development of various software products, applications and information systems developed for government and commercial sectors as well as for common users there is the need to write a large amount of technical documentation describing the created product from various perspectives. The reasons for such documentation – «internal» and «external» ones are presented in this paper. The types of documentation: requirements documentation (defines the expectations for the software, including functional requirements, hardware requirements, compatibility and limitations); architecture documentation (defines the high-level architecture of the system: the components, their functions and the data and control flow); technical documentation (written for technical audience, describing the code, algorithms and interface); user manuals (there are several types of user documentation: tutorial, how-to guide and reference guide); marketing documentation are described. Documentation standards (free and paid) are given as well.*

***Key words**: documentation, software, software documentation, technical requirement, documentation standards, documentation requirements, user manuals, marketing documentation, architecture documentation.*

https://doi.org/10.33108/visnyk_tntu2020.02.120 *Reseived 17.04.2020*

Statement of the problem. At present dozens and hundreds of different programs, applications, information systems appear every day. They are developed for both government and commercial sectors as well as for ordinary users. Software development (SW) is sufficiently complex, long and expensive process, requiring a large amount of technical documentation containing the description of created product from various perspectives. Technical documentation can include not only the basis for development and maintenance guide of the off-the-shelf software product, but also other artifacts created at different development stages.

Therefore, there is the need to create qualitative documentation for software which meets all the requirements and standards throughout the project life cycle (LC). It is also necessary to investigate and determine the types of documentation during SW development.

Analysis of available investigations results. The scientists, such as V. Ye. Shikina [1], O. H. Iniushkina, M. N. Krasnianskiy [2], A. V. Ostroukh and other national scientists pay much attention to the analysis of technical documentation for information systems.

In their scientific papers N. Ye. Surkova [3], V. M. Gurianov and others dealt with the problem of analysis of activities concerning the development of international and national standards in the field of modern information technology, they considered the problem of interaction between standards and modern methodologies for complex software development, and presented the list of documents required by the standard and requirements to the documents recording at all stages of software development.

Scientists Martin Fowler, Robert C., Donald Knuth, Steve Krug and others, paid much attention to the creation and implementation of software and documentation.

Due to the emergence of new technologies and few investigations in terms of software documentation, taking into account the standards, there is the need for further research and improvement of project documentation.

Corresponding author: Anna Synko; e-mail: anna46075@gmail.com

The objective of the paper is to determine the need of software documenting, separation of documents types, requirements and standards.

Presentation of basic material. Documentation is an integral part of project creation [4]. Each software has its life cycle – a period of time starting with the decision making concernig software product creation and ending at the time of its complete retirement [5].

For easy design, creation and implementation of qualitative software product there are various SW life cycle models [6]. The most widely used ones are:

- Cascade or waterfall model;
- v-model;
- incremental model;
- spiral model;
- agile model;
- evolution prototyping model.

Each of these models has its own requirements for the system, the sequence of software development stages execution, as well as advantages and disadvantages that can be eliminated by using another model [7]. There are some differences in determining the number of phases and content of the selected model as these characteristics depend on the task, conditions of particular project execution and experience of the involved participants.

The main stages of the project life cycle (LC) include – planning, analysis, design, development, testing, implementation, operation [8]. Each of these should be followed by documentation. Software documentation are printed user manuals, dialog (operational) documentation and reference text describing how to use the software product [9]. Documentation contains documents – elements of documentation: target information intended for specific audience, located on specific medium (for example, in the book, on disk, in brief reference map) in the given format [9].

Software documentation are documents that, depending on the purpose, contain the data necessary for the development, production, operation, maintenance of the program or software [10].

First let us define the purpose of documentation while developing software:

1) Documentation provides the «common space» of the project. Any participant at any time can get the necessary information on the specific task and on the general direction of the work as well.

2) The team speaks «one language» as it is much easier to understand a person who reports, for example, the error in the described function by means of Use Case [11] than to report the problem described in the language (which can contain its own special vocabulary) of a certain person.

3) Documentation makes it possible to distinguish clearly the areas of responsibility between project participants.

4) Only carefully described requirements can be checked for completeness and consistency. «Knee notes» are direct and very fast way to the fact that the project boundaries will extend very quickly, and the planned functionality, will not be implemented with originated customer wishes and requirements.

All of these are «internal» reasons for the need of documentation, but there are also external reasons for documentation creation, which belong to the following types: user and marketing documentations, they are listed below.

Therefore, the importance of documentation during project implementation is defined. Now, for better uderstanding of documentation formation, let's describe this process from the very begining.

Upon project receipt from the customer, the technical task (TT) is formed, it contains: the dictionary of terms for the subject area; description of the subject area; description of the role system; description of functional requirements; description of non-functional requirements – all this refers to the type of documentation – requirements. Further all other types of documentation are formed.

The requirements description in the document is fixed at the «upper level», i.e. not the description of specific actions, but only necessary functions [12]. Requirements are optimally divided into semantic groups by subsystems (Figure 1).

If the system is large, it is reasonable to make «separate technical tasks» (STT) for each subsystem [13]. STT should contain the following: reference to TT section; maximum detailed information on each function; Use Case list for function. Thus, the documents consistency is realized, this makes it possible, firstly, to unify their form, and secondly – to implement partially the reuse, i. e. to reduce the time spent on writing the same type of documentation.

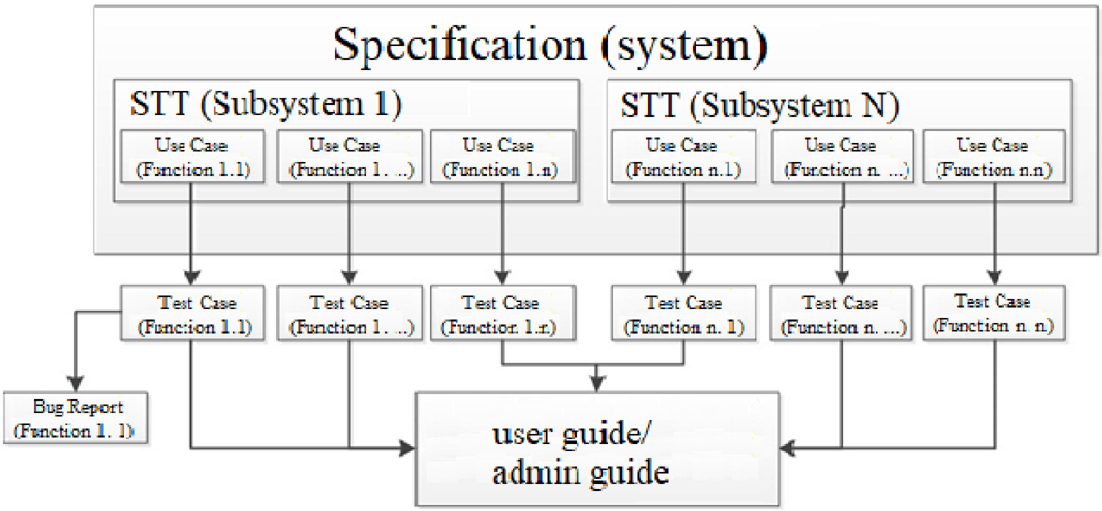

Figure 1. The example of the general scheme for documents linking

In order to design such schemes during software development Use Case tools (the essence of the use case) are usually applied. Due to these tools you can describe all the actions that the user can do, and the system response to these actions [10]. Each Use Case should be linked to STT section.

The Test Case should contain the test scenarios description [14]. Ideally, each such document is bound to the corresponding Use Case, but sometimes several Use-Cases can be logically combined into one Test Case [15]. The best option for the description format is the table containing in one column the description of atomic operation, which responds to the system action, in the second column – the description of correct system reaction. For example, you do not need to describe the process of entering the text into the text box, but you should check the data validity while storing (pressing «Save» button).

Also, while designing the system, it is important to create «Bug Report», which occurrs in the process of system testing as the response of the tester to the error. Each document should refer to the appropriate Test Case. This type of document should contain: screenshot of error occurrence; description of the previous actions (it is better to develop the convenient template for such description – it significantly reduces the time spent by developers when reproducing the error); text description of the error itself.

Therefore, after presentation of the general scheme of technical task it is necessary to separate and think about registration of such basic types of documentation for software:

- requirements documentation – the statements defining the system attributes, capabilities, characteristics and properties. This is the basis for implementation;
- architectural/design documentation – the software review, including the description of the working environment and the principles that should be used when creating software;
- technical – documentation for code, algorithms, program interfaces;
- user-friendly documentation – manuals for end users, system administrators and the staff;
- marketing.

Let us give the detailed description of each type of documentation:

Documentation requirements are the description particular software action. The requirements are created and used by all participants involved in software development and operation: end users, customers, software products managers, managers of project, sales, marketing, software architects, engineers, interaction designers, developers and testers. Thus, the documentation requirements have many different purposes.

Documentation requirements are provided in various styles, notations and formalities. They can be represented in the form of goals, mathematical figures and formulas, as well as in combination with each other.

The requirements can be implicit, which complicates the task. It is difficult to know exactly how many and what documents are needed and how much documentation is needed for architecture and design, and it is difficult to understand how to document requirements taking into account the diversity of people who will read and use the documentation. Thus, the documentation requirements are often incomplete (or non-existent). Without proper documentation, software changes are difficult – and therefore more error-prone (resulting in low software quality) and time-consuming (expensive for the project).

Table 1

Summary documentation requirements

Program documentation requirements	
describe	what specific software does or will do
used	as agreement or as basis for the agreement on what the software will do
created and applied by	end users, customers, product managers, managers of project, sales, marketing, software architects, engineers, interaction designers, developers and testers, etc.
presented	in different styles, notations and formalities

The need for documentation requirements is usually related to the product complexity, product influence and the length of software LC.

Based on the above mentioned let us define the following basic requirements for documentation:

- documents should be accurate, complete and, if possible, brief with clear and unambiguous interpretation [1];
- documentation should be created in parallel with the development of the information system itself;
- the developer is responsible for system documenting;
- standards governing the form and content of documents should be used in order to increase the efficiency of document management.

The application of such requirements will help in the documentation formation – will improve efficiency and save financial and labor resources, time.

Architectural/design documentation usually describes the product in general terms. It does not describe how and what is used, but this type of documentation answers the question «why exactly so». For example, in project document, the programmer can describe the rationale why data structures are organized in this way. The reasons for such class design are described in, patterns (templates, samples) are distinguished, in some cases even the ideas how to improve the software product or its separete functions in the future are given. None of the above mentioned is included in technical or user documentation, but it is an important part of the project.

Technical documentation. While creating the program, only code is usually not enough. A certain text describing different aspects of code operation is needed. Such documentation is often included directly in the output code or provided along with it.

Such documentation has strong technical nature and is used mainly to define and describe API, data and algorithms structures.

Automated tools such as Doxygen, javadoc, NDoc and others are often used in technical documentation development. They receive information in special way from the comments in the output code, and create reference guides in any format, such as text or HTML [16].

The use of comment-based documentation generators in program code is considered to be the convenient tool for many developers, and is caused by variety of reasons. Particularly, in such approach, the documentation is the part of the output code, and the same tools can be used to write the program and collect documentation for it at the same time. It also simplifies keeping the documentation up to date.

User documentation. Unlike technical documentation focused on code and how it works, user documentation describes only how to use the program.

If the product is a software library, user documentation and code documentation become very close, almost equivalent concepts. But in general, this is not the case.

Typically, this type of documentation is the user guide describing each function of the program, as well as the steps that should be performed for this function use [11]. Excellent, high-quality documentation for the user also provides instructions what to do in case of problems [17]. It is very important for the documentation not to be misleading and be relevant. The reader guide should have clear structure; it is very useful if there is through subject index. Logical connectivity and simplicity are also important, as this type of documentation is intended for a wide range of people.

There are three approaches user documentation management:

- tutorial, the most useful for new users. It provides detailed description of the steps for typical task execution;
- thematic approach, in which each section is devoted to the specific topic. It is more suitable for users who improve their skills or have good understanding of the actions of other operations related to the selected one;
- in this approach, tasks or commands are organized in the form of the alphabetical guide – this is usually suitable for experienced users who know exactly what they need to look for.

The use of such documentation management, divided into classes of users, makes it more expensive, but better and more convenient from the point of view of users. In addition, complaints from users usually relate to the fact that the documentation covers only one of these approaches, and therefore is well suited only for one class of users, but not for a wide range.

In most cases, software developers limit the set of user documentation by built-in assistance system containing information about commands or menu items. Training and supporting new users is provided by private publishers, who usually significantly assist the developers.

Marketing documentation. Most of the created projects require advertising materials in order to interest people by drawing their attention to the product [18]. This form of documentation is intended to:

- to stimulate potential users interest in the product;
- to inform about exact product actions, compared to the fact that consumers expectations coincide with that what they receive;
- to explain the product position in comparison with competing solutions.

One of the good marketing practices is to provide the slogan – a simple phrase easy to remember and which demonstrates what we want to bring to the user, as well as characterizes the feeling created by product (or about the created product) [19]. After all, it often happens that the product box and other marketing materials give more exact picture of the possibilities and ways of using the program than anything else.

Marketing documentation is usually used for such programs, which should be:

- known and accessible to a wide range of users;
- usually have (contain) entertainment or product-service functions.

After all, software, for example, for internal operation of military unit is forbidden for advertising and distribution, as it can contain confidential information, the declassification of which will harm people lives.

Let us present the main marketing areas for software promotion:

- SEO;
- PPC advertising (Google, Begun etc.);
- software localization;
- advertising in blogs;
- corporate blog;
- activity concerning the “ease of use” of the site;
- marketing techniques on the site.

Documentation standards.

While cooperating with customers, it is always necessary to submit a certain package of documents – manuals, instructions, design solutions, issued in accordance with the requirements of regulatory documents.

We should not neglect the fact that while managing the documentation the latter must be drawn up in accordance with the requirements and standards of regulatory documents [3], [20]. There are free standards (for example, GOST 34.602-89 «Technical task for automated system creation» [21] – the standard for TT) and paid (for example, IEEE Std 1063-2001 «IEEE Standard for Software User Documentation» (the document indicates the requirements for structure, content and format of user manuals), IEEE Std 1016-1998 «IEEE Recommended Practice for Software Design Descriptions» [22] (recommendations for documents describing the software architecture, or the technical description pack), ISO/IEC FDIS 18019: 2004 «Guidelines for the design and preparation of user documentation for application software», ISO/IEC 26514: 2008 «Requirements for designers and developers of user documentation» etc.).

The above mentioned standards are most closely related to documentation.

Conclusions. Due to the rapid development of various software products, applications and information systems there is the need to write a large amount of technical documentation describing the created product from various perspectives. As any software has life cycle (the most common are: cascade or waterfall, incremental, spiral models, etc.), each stage of which contains certain documentation, documentation becomes an integral part of the project creation and development. «Internal» and «external» reasons explaining its importance and necessity are given.

For better understanding where the documentation is formed, the process is described from the very beginning – receiving the project from the customer and forming the technical task. After that the need to develop such types of documentation which covers all areas, aspects of the project, occurs. This documentation includes: requirements documentation, architectural/design, technical, user and marketing documentation. Detailed, extended description of each type is provided for better understanding of what should to be done at each documentation stages. When documentation is carried out in accordance with these types, the software documentation is said to be complete and holistic.

Free and paid standards of documentation application of which helps to issue all necessary materials correctly (according to requirements of regulatory documents) are given.

The chosen direction of our investigation is important and requires further development, because new occurring technologies for creation, formation, design, optimization of software documentation, etc., are to be mastered and involved in the documentation process.

References

1. Shikina V. E. Tekhnicheskaya dokumentatsiya informatsionnykh sistem: uchebnoye posobiye. Ulyanovsk: UlSTU, 2018. 92 p. [In Russian].
2. Krasnyanskiy M. N. and others Proyektirovaniye informatsionnykh sistem upravleniya dokumentooborotom nauchno-obrazovatelnykh uchrezhdeniy: monograph. Tambov: TSTU 2015. 216 p. [In Russian].
3. Surkova N. E., Gurianov V. M. Rukovodstvo po ispolzovaniyu standartov pri razrabotke slozhnykh programmnykh sredstv: ucheb. posobiye. Moscow, MADI, 2018. 92 p. [In Russian].
4. Pleskach V. L. Zatonats'ka T. H. Informatsiyni systemy i tekhnolohiyi na pidpryyemstvakh: pidruchnyk. Kiev, Znannia Publ., 2011. 718 p. [In Ukrainian].
5. Komova M. V., Peleshchyshyn A. M., Bilushchak T. M. Keruvannya dokumentatsiynymy protsesamy, Lviv: Publisher: Lviv polytechnic national university, 2013. [In Ukrainian].
6. Aleksenko O. V. Tekhnolohiyi prohramuvannya ta stvorennya prohramnykh produktiv: konspekt lektsiy dlya stud. napryamu pidhotovky 6.050101 “Computer science” all forms. Sumy: Sumy State University, 2013. 133 p. [In Ukrainian].
7. Peleshchyshyn A. M., Trach O. R. Vyznachennya elementiv sotsial'no-oriyentovanykh ryzykiv pry orhanizatsiyi zhyttyevoho tsyklu virtual'noyi spil'noty. Ukrainian Scientific Journal of Information Security, 2017, no. 23, vol. 2, 130–135 pp. [In Ukrainian].
8. Valacich, Joseph S., George, Joey F. Modern Systems Analysis and Design (8th Edition). Publisher: Pearson, 2017, 545 p.
9. GOST R ISO/MEK 15910-2002 – Protsess sozdaniya dokumentatsii polzovatelya programmnogo sredstva. Moscow, IPK Standards Publishing. 2002. [In Russian].
10. Laktionov Ye. Yu., Malyts’kyy A. Yu. Vykorystannya XML dlya pidhotovky spetsializovanoyi prohramnoyi dokumentatsiyi. National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute” Series: Informatics, control and computer engineering. 2010, no. 52, 131–136 pp. [In Ukrainian].
11. GOST 19781–90 Software of data processing systems. Terms and definitions. Introduced on 01.01.92. Moscow, Gosstandart of Russia: Publishing House of Standards, 1992, 14 p. [In Russian].
12. Shaposhnyk S. P. Upravlinnya dokumentatsiyeyu v orhanizatsiyi. Kharkiv Regional Institute of Public Administration: State building, 2008, no. 64, vol. 1 [in Ukrainian].
13. Zhyryakova I. A., Hadets'ka Z. M. Metodolohiya funktsional'noho modelyuvannya yak zasib predstavlennya proektnoyi dokumentatsiyi v IT-aut·sorsinhu. Kharkiv: National Technical University “Kharkiv Polytechnic Institute”, 2010, 122–128 pp. [in Ukrainian].
14. Kovalenko I., Shved A., Davydenko Ye. (2019) Choice of software development technologies based on pareto-optimal solutions. Scientific Journal of TNTU (Tern.), vol. 95, no. 3, pp. 116–122. https://doi.org/10.33108/visnyk_tntu2019.03.116.
15. Bouraou N., Tsybulnik S., Rupich S. (2017) Problems of Intellectualizing in SHM Systems: Estimation, Prediction, Multi-Class Recognition. Scientific Journal of TNTU (Tern.), vol. 88, no. 4, pp. 135–144. https://doi.org/10.33108/visnyk_tntu2017.04.135.
16. Sushchyk A. M. Metod stvorennya dokumentatsiyi dlya REST API na osnovi testiv: master's thesis for the second level of higher education. Kiev. National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”, 2019. 130 p. [In Ukrainian].
17. Lavrishcheva E. M. Software Engineering kompyuternykh sistem. Paradigmy. tekhnologii i CASE-sredstva programmirovaniya. Kiev, Naukova Dumka Publ., 2013, 283 p. [In Russian].

18. Smychenko O. S., Systema dlya poshuku ta optymizatsiyi reklamnykh kampaniy: bachelor's degree to obtain the first level of higher education. 467200.001. Kiev, 2019. 82 p. [In Ukrainian].
19. Fedorovych R. V. ta in. Marketynhovyy instrumentariy upravlinnya popytom na tovary i posluhy / edited by prof. R. V. Fedorovycha. Ternopil: Textbooks and manuals, 2016. 244 p. [In Ukrainian].
20. Antonenko I. Do istoriyi rozroblennya mizhnarodnoho standartu ISO 15489-2001 "Information and documentation. – Document management". Archival science. Archaeography. Source studies. Kiev. 2003. No. 6. 73–83 pp; Draft concept of implementation of international standard norms in Ukraine ISO 15489-2001 "Information and documentation. – Document management". Studies with arch. cases and documentation. Kiev, 2004, 172–177 pp [in Ukrainian].
21. GOST 34.602-89. Information technology. Set of standards for automated systems. Technical directions for automated system making. Moscow, STANDARTINFORM. 2009. [In Russian].
22. IEEE Std 1016-1998. IEEE Recommended Practice for Software Design Descriptions. IEEE-SA Standards Board 1998. 16 p.

Список використаної літератури

1. Шикина В. Е. Техническая документация информационных систем: учебное пособие. Ульяновск: УлГТУ, 2018. 92 с.
2. Краснянский М. Н. Проектирование информационных систем управления документооборотом научно-образовательных учреждений: монография. Тамбов: Изд-во ФГБОУ ВПО «ТГТУ», 2015. 216 с.
3. Суркова Н. Е., Гурьянов В. М. Руководство по использованию стандартов при разработке сложных программных средств: учеб. пособие. М.: МАДИ, 2018. 92 с.
4. Плескач В. Л., Затонацька Т. Г. Інформаційні системи і технології на підприємствах: підручнику К.: Знання, 2011. 718 с.
5. Комова М. В., Пелещишин А. М., Білущак Т. М. Керування документаційними процесами. Львів: Вид-во НУ «Львівська політехніка», 2013.
6. Алексенко О. В. Технології програмування та створення програмних продуктів: конспект лекцій для студ. напряму підготовки 6.050101 «Комп'ютерні науки» усіх форм навчання. Суми: СумДУ, 2013. 133 с.
7. Пелещишин А. М., Трач О. Р. Визначення елементів соціально-орієнтованих ризиків при організації життєвого циклу віртуальної спільноти. Науковий журнал «Безпека інформації». 2017. Вип. 23. № 2. С. 130–135.
8. Valacich, Joseph S., George, Joey F. Modern Systems Analysis and Design (8th Edition). Publisher: Pearson, 2017. 545 p.
9. ГОСТ Р ИСО/МЭК 15910-2002: Процесс создания документации пользователя программного средства. М.: ИПК Издательство стандартов, 2002.
10. Лактіонов Є. Ю., Малицький А. Ю. Використання XML для підготовки спеціалізованої програмної документації. Вісник НТУУ «КПІ». Сер.: Інформатика, управління та обчислювальна техніка. 2010. Вип. 52. С. 131–136.
11. ГОСТ 19781–90: Единая система программной документации. Обеспечение систем обработки информации программное. Введ. 01.01.92. М.: Изд-во стандартов, 1992. 14 с.
12. Шапошник С. П. Управління документацією в організації. Державне будівництво. 2008. Вип. 64. № 1.
13. Жирякова І. А., Гадецька З. М. Методологія функціонального моделювання як засіб представлення проектної документації в ІТ-аутсорсінгу. Харків: КПІ, 2010. С. 122–128.
14. Kovalenko I., Shved A., Davydenko Ye. Choice of software development technologies based on pareto-optimal solutions. Scientific Journal of TNTU. 2019. Vol 95. No. 3. P. 116–122. https://doi.org/10.33108/visnyk_tntu2019.03.116.
15. Bouraou N., Tsybulnik S., Rupich S. Problems of Intellectualizing in SHM Systems: Estimation, Prediction, Multi-Class Recognition. Scientific Journal of TNTU. 2017. Vol. 88. No. 4. P. 135–144. https://doi.org/10.33108/visnyk_tntu2017.04.135.
16. Сущик А. М. Метод створення документації для REST API на основі тестів: магіст. р. на здобуття другого рівня вищої освіти. Київ. КПІ ім. І. Сікорського, 2019. 130 с.
17. Лаврищева Е. М. Software Engineering компьютерных систем. Парадигмы, технологии и CASE-средства программирования. К.: Наук. думка, 2013. 283 с.
18. Смиченко О. С., Система для пошуку та оптимізації рекламних кампаній: бакалавр. на здобуття першого рівня вищої освіти. 467200.001. Київ, 2019. 82 с.
19. Федорович Р. В. та ін. Маркетинговий інструментарій управління попитом на товари і послуги / за ред. проф. Р. В. Федоровича. Тернопіль: Підручники і посібники, 2016. 244 с.

20. Антоненко І. До історії розроблення міжнародного стандарту ISO 15489-2001 «Інформація та документація. Управління документацією». Архівознавство. Археографія. Джерелознавство. 2003. Вип. 6. С. 73–83.
21. Антоненко І. Проект концепції впровадження в Україні норм міжнародного стандарту ISO 15489-2001 «Інформація та документація. Управління документацією». Студії з арх. справи та документознавства. К., 2004. Т. 11. С. 172–177.
22. ГОСТ 34.602-89. Техническое задание на создание автоматизированной системы. М.: Стандартинформ, 2009.
23. IEEE Std 1016-1998. IEEE Recommended Practice for Software Design Descriptions. IEEE-SA Standards Board 1998. 16 p.

УДК 004.91

ДОКУМЕНТУВАННЯ ПРИ РОЗРОБЦІ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ – ТИПИ ТА СТАНДАРТИ ДОКУМЕНТАЦІЇ

Анна Синько; Андрій Пелещишин

Національний університет «Львівська політехніка», Львів, Україна

Резюме. *Через стрімкий розвиток різноманітних програмних продуктів, додатків, інформаційних систем, що можуть бути розроблені як для державного або комерційного сектора, так і для звичайних користувачів, виникає потреба у написанні великої кількості технічної документації, що містить опис створюваного продукту з різних точок зору. Зрозуміло, що кожне програмне забезпечення має свій життєвий цикл, який реалізується за допомогою моделей (до найпоширеніших з яких належать: каскадна або водоспадна, інкрементна, спіральна моделі тощо) та етапів, кожен з яких повинен супроводжуватися документацією. В роботі наведено для чого взагалі потрібна документація – «внутрішні» та «зовнішні» причини. Тепер, для кращого уявлення звідки взагалі формується документація, було описано процес спочатку при отриманні проекту від замовника, і, надалі, побудовано загальну схему організації зв'язку між документами, для проектування якої рекомендується застосовувати Use Case засоби. Після побудови такої схеми необхідно провести документування відповідно до типів документації: документування вимог (необхідні для визначення атрибутів, можливостей, характеристик та властивостей системи тощо, а також вони є описом того, що конкретне програмне забезпечення робить або буде робити), архітектурна/проектна (наведено огляд програмного забезпечення, що включає опис робочого середовища і принципів, які повинні бути використані при створенні програмного продукту), технічна (надає певний текст, що описує різні аспекти того, що саме робить програмний код. Подібна документація має сильно виражений технічний характер і в основному використовується для визначення й описування API, структур даних і алгоритмів за допомогою автоматизованих засобів, генераторів), користувацька (містить керівництво для кінцевих користувачів, адміністраторів системи та іншого персоналу. Рекомендовано застосовувати три підходи до організації даного типу документації) та маркетингова документації (призначена для рекламування, поширення, розповсюдження програмного продукту за допомогою сучасних інструментів просування). Також у роботі наведено стандарти (безкоштовні та платні), застосування яких робить документацію повною та якісною (відповідно до вимог нормативних документів).*

Ключові слова: *документація, програмне забезпечення, програмна документація, технічне завдання, стандарти документації, вимоги до документації, користувацька документація, маркетингова документація, архітектурна/проектна документація.*

https://doi.org/10.33108/visnyk_tntu2020.02.120 *Отримано 17.04.2020*