UDC 001.8

Siberian Journal of Science and Technology. 2017, Vol. 18, No. 4, P. 706–710

RESEARCH INTO THE METHODS OF SOFTWARE PRODUCT DEVELOPING AND MAINTAINING

Zh. S. Abenova[1*], M. N. Petrov[2]

[1]JSC "National Company "Kazakhstan Gharysh Sapary"
National Space Centre, 89, Turan Av., Astana city, 010000, The Republic of Kazakhstan
[2]Reshetnev Siberian State University of Science and Technology
31, Krasnoyarsky Rabochy Av., Krasnoyarsk, 660037, Russian Federation
[*]E-mail: zhuza44@mail.ru

Currently the development of information and Open Source technologies for the implementation of critical business functions in fields of national economy, including space industry becomes relevant. However, as experience shows there is no uniform method for the process of developing and operating the software product using Open Source technology. The aim of the article is to study classical and modern models and technologies of software analysis and design. It will help choose the optimal model for the development of the method for creating a prototype information system using Open Source technology, and also to define the design environment and the tasks of implementing the prototype of the information system. The article considers a summary table which allows to choose efficient model for developing the method of building the information system, taking into account the specifics of the free software products.

Keywords: software, information system, model software development.

Сибирский журнал науки и технологий. 2017. Т. 18, № 4. С. 706–710

ИССЛЕДОВАНИЕ МЕТОДОВ ПРОЦЕССА РАЗРАБОТКИ И ЭКСПЛУАТАЦИИ ПРОГРАММНОГО ПРОДУКТА

Ж. С. Абенова[1*], М. Н. Петров[2]

[1]АО «Национальная компания «Қазақстан Ғарыш Сапары»
Казахстан, 010000, г. Астана, просп. Туран, 89, Национальный космический центр
[2]Сибирский государственный университет науки и технологий имени академика М. Ф. Решетнева
Российская Федерация, 660037, г. Красноярск, просп. им. газ. «Красноярский рабочий», 31
[*]E-mail: zhuza44@mail.ru

В настоящее время является актуальным развитие информационных технологий с применением технологий Open Source для реализации критически важных бизнес-функций в сферах народного хозяйства, в том числе космической отрасли. Однако, как показывает опыт, отсутствует единая методика процесса разработки и эксплуатации программного продукта с применением технологий Open Source. Целью статьи является изучение классических и современных моделей и технологий анализа и проектирования программных средств. Это позволит выбрать оптимальную модель для разработки методики по созданию прототипа информационной системы, использующую технологии Open Source, а также определить среду проектирования и задачи реализации прототипа информационной системы. Рассматривается сводная таблица, которая позволяет выбрать оптимальную модель для разработки методики построения информационной системы с учетом специфики свободных программных продуктов.

Ключевые слова: программное обеспечение (ПО), информационные системы, модель разработки ПО.

Introduction. Information technologies are one of most dynamic developing areas of science, technology and engineering, besides they are included in the list of critical technologies, which contributes to the improvement and continuity of new solutions, aimed at creating advanced information systems [1]. Therefore, the application of an integration environment of information cooperation (IEIC) based on Open Source technology in the aerospace segment is an actual solution that will solve the issues of information support in managing projects of various complexity and direction, regardless of the subject's location. IEIC system is the cheapest one and has all necessary functions and capabilities, taking into account tight budget and human resource constraints. In this respect, in order to form the information environment of the IEIC it is necessary to investigate the basic strategies and their models for the process of developing and putting into operation a software product. This will enable to choose the most effective design model of the IEIC system development, to define the design environment and to describe the project characteristics, the concept and the tasks of implementing the IEIC prototype [2].

Analysis of the efficient method selection for the IEIC system creation and development.
According to the Russian software development standards, there are three basic strategies: cascading, incremental and evolutionary models [3]. The cascading strategy is a linear sequential design approach for software development; the incremental strategy assumes that the requirements for the software product are implemented gradually, each time expanding a product utility; in the evolutionary strategy, the requirements are not fully defined, but are dynamically refined during the development of software product versions [4; 5].

Figure shows basic models of the basic software development strategy.

However, although there are many different models and ways of software products development, there is no single method that describes the Web resource development model using Open Source technologies, such as Development of IEIC. Therefore, to select a life cycle model Software Quality Institute in USA recommends to classify the project and to identify the main parameters that will be the key factor to select the model of software development [6]. These include:

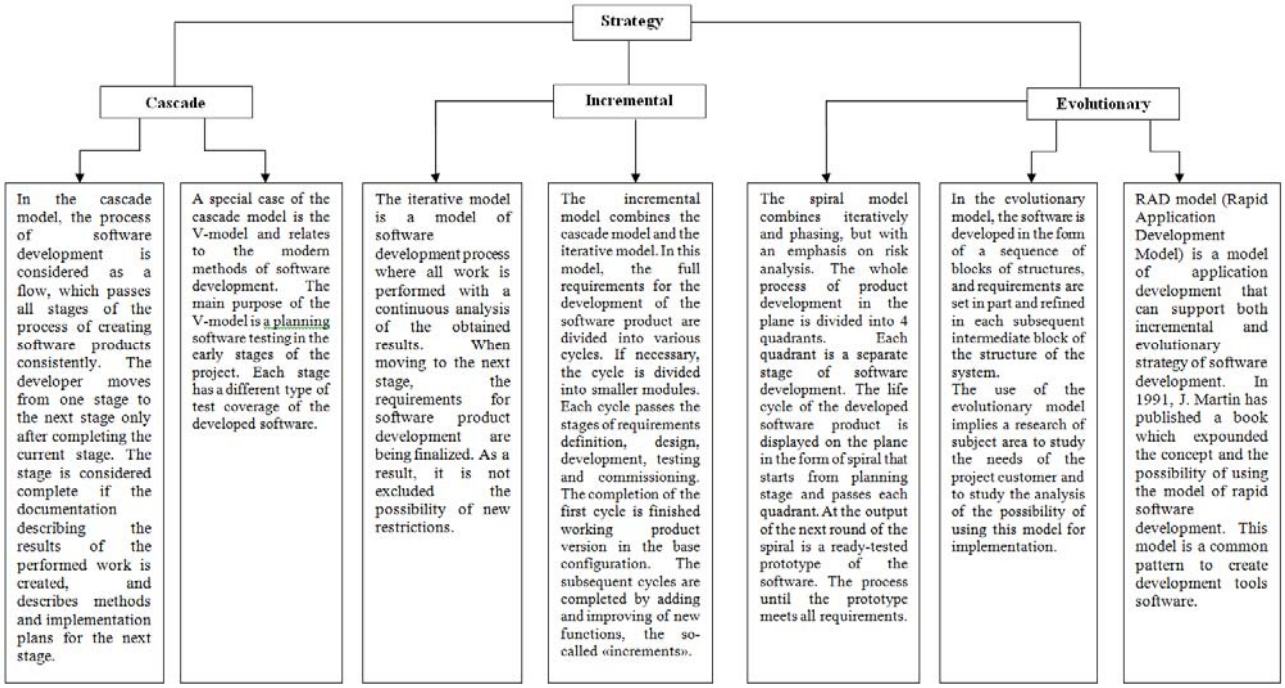
1. Requirement parameters for the project. This category can include the possibility to set up requirements for software at the beginning of the software life cycle, to determine the probability of changes and additions to the requirements for the software development lifecycle.

2. Parameters of the project working team. This category determines the novelty of technologies and the novelty of toolbox development for developers.

3. User profiles. This category identifies the degree of users involvement in the development process and their relationship to the project team, determines the degree of user involvement in the process of developing software and acquaintance of users with the problems in the process of the software life cycle.

4. Parameters of project types and risks. This category reflects the complexity of the project, estimates the resources for its execution, identifies problems in the software domain and determines requirements for reliability levels and other criteria [7].

Based on the above said a summary table that includes all categories of project parameters is suggested. Table will help to choose suitable model of software lifecycle for a particular project [8–10].



Software development models

Models of software engineering

№	Classification parameters	Model development of life cycle of the software product						
		Cascading	V-model	Iterative	Incremental	Spiral	RAD	Evolutionary
1	Are requirements for the project defined and feasible?	Yes	Yes	Yes	Yes	No	Yes	No
2	Can requirements for the project be determined at the beginning of the software life cycle?	Yes	Yes	Yes	Yes	No	Yes	No
3	Do you need to demonstrate requirements for clear understanding?	No	No	No	No	Yes	Yes	Yes
4	Is it necessary to check the concept of the software product?	No	No	No	No	Yes	Yes	Yes
5	Can the requirements be changed or added in the process of the software life cycle?	No	No	Yes	Yes	Yes	No	Yes

End table

№	Classification parameters	Model development of life cycle of the software product						
		Cascading	V-model	Iterative	Incremental	Spiral	RAD	Evolutionary
6	Is there a need to implement requirements in the early stages of software development?	No	No	No	Yes	Yes	Yes	Yes
7	Are the tasks of the subject area of the project new to the project team?	No	No	Yes	No	Yes	No	Yes
8	Are the tools for the implementation of the project new to the project team?	No	No	Yes	No	Yes	No	No
9	Can participants change roles in the process of software lifecycle development project?	No	No	Yes	Yes	Yes	No	Yes
10	Is in the process of the software lifecycle evolution and validation stage of software development needed?	Yes	Yes	No	Yes	Yes	No	Yes
11	Will users participate in the process lifecycle?	No	No	Yes	Yes	Yes	No	Yes
12	Will users evaluate the current status of the software in the process of software development?	No	No	Yes	Yes	Yes	No	Yes
13	Will users be involved in all phases of the software life cycle development?	No	No	Yes	No	Yes	Yes	No
14	Will users track the progress of the project?	No	No	Yes	No	Yes	No	Yes
15	Is the software being developed new to the company?	No	No	Yes	Yes	Yes	No	Yes
16	Will the project be an extension of the existing system?	Yes	Yes	No	Yes	Yes	Yes	No
17	Will the project be large-scale?	No	No	Yes	Yes	Yes	No	Yes
18	Will the project be medium or small-scale?	Yes	Yes	No	No	No	Yes	Yes
19	Will the product life cycle be long-lasting?	Yes	Yes	Yes	Yes	Yes	No	Yes
20	Is a high level of product reliability necessary?	No	Yes	Yes	Yes	Yes	No	Yes
21	Is it planned to upgrade and develop the product during the operation phase?	No	No	No	Yes	No	No	Yes
22	Is the timetable tight?	No	No	No	Yes	No	Yes	Yes
23	Will the functions and modules be reused?	Yes	No	No	Yes	No	Yes	Yes
24	Is the project budget tight?	No	No	No	No	No	No	Yes
25	Are the software developers sufficiently competent?	Yes	Yes	Yes	Yes	Yes	Yes	Yes
26	Can costs be required to purchase additional equipment and tools for project implementation?	No	No	Yes	Yes	Yes	Yes	Yes
27	Can additional costs be required to attract highly qualified employees?	No	No	Yes	No	Yes	Yes	Yes
28	Have such software products been implemented in the enterprise before?	Yes	No	No	No	No	Yes	No
29	Will new technologies or new approach be used in the development process?	No	Yes	Yes	Yes	Yes	No	Yes
30	Is it necessary to conduct analysis of existing technologies for software development?	No	No	Yes	Yes	Yes	No	Yes
31	Is it necessary to develop a prototype of the software product in the early stages of the product life cycle?	No	Yes	No	Yes	Yes	Yes	Yes
32	Is the software unique?	No	No	Yes	No	Yes	No	Yes
33	Is the software part of the system?	Yes	Yes	No	No	Yes	No	No
34	Is the software a stand-alone solution?	No	No	Yes	Yes	No	Yes	Yes

According to table the cascade model is more suitable when the requirements are known, straightforward and documented, and there are no problems with the availability of programmers of necessary qualifications [11–13]. This model can be used in relatively small projects, in the redevelopment of typical software (for example, the development of an electronic document management, accounting system, etc.). The cascading model can be used in case of migrating of existing software to a new platform or when launching a new version of software if the changes are minor and clearly defined. The advantage of this model is that each stage is completed with certain documentation that meets the criteria of completeness and consistency. Therefore, there is no need to return to the previous stages. All work is performed in strict order. This approach allows to schedule deadlines, and accordingly plan the costs of each stage in advance. The disadvantage of the model is the complexity of switching between stages [14].

The V-model fits better when requirements are defined and documented, thorough testing is required. The model is suitable for medium and small-scale projects,

provided that qualified programmers, including testers are involved. The advantages of this model include requirements setting prior to development, tests planning and system verification at the early stages of software development, special importance is given to time management. However, there are disadvantages: the model does not imply the work with parallel events; there is no possibility to introduce requirements for dynamic changes at different stages of the software life cycle. The model can be used mainly in projects with tight time or financial constraints, also in such projects where there is an extensive coverage of the test tasks.

The iterative model can be used for large-scale projects, when the requirements for the final system are clearly defined and understood, but can be modified over time. That is, the model implies that the main task must be defined, and implementation details may change during the development of the software product over time.

The incremental model is the most suitable when basic requirements for the system are clearly defined and if the software needs early introduction on the market, however, there is a possibility of revision over time. This model can be used in long-term projects at low or medium risks. Also, the model is suitable for projects where new technology can be applied. This will allow the user to adapt to the software product being developed by executing smaller incremental steps at specific time intervals. The advantage of this model is that the result of each increment is the working functional product and a customer has the opportunity to evaluate each developed version of the software. Firstly, it is supposed to develop and implement the main function of the platform in the model and only then increase it. This will reduce the risks of failures and risks of over-expenditures, as the main functionality of the system can be determined at the early stages of development. The disadvantages of the model include the fact that at the initial stage it can be difficult to define a complete efficient system which makes it difficult to make up the increments.

The spiral model is suitable for complex and expensive projects, where it is required to carry out analysis for impact evaluation after each step. If software development involves the use of new technology and success is not guaranteed, then the spiral model should also be used. This model is appropriate for development of new series of software products, where it is important to analyze the risks and costs. In the model it is possible to return to the left behind stages to reflect over the changes in costs of risks. The advantages of the model include step by step specification requirements and conducting a risk analysis of the project, which allow to identify design errors at the early stages of software development, creation of working prototypes at the early stages and availability of completed software development process documentation, as well as the ability to add new features even at the later stages of the software life cycle. A complex project can be developed in parts, highlighting the most important requirements at different stages. The disadvantages of the spiral model include the dependence of success of the development process on the stage of risk analysis and high cost of software development, because risk management requires the involvement of sufficiently qualified specialists.

The RAD model can be used only in the presence of highly qualified specialists as this model is suitable for confident knowledge of the target business and the need for urgent production of system [15]. This model is appropriate for projects where the budget is large, in order to pay for professional service. Also, this model can be used in projects where there is very little risk; the software product being developed can be modeled and has relatively low performance; and when it is required to minimize duration of software product development. The advantages of this model include reduction of the cycle time and the number of developers by using powerful tools. Already in the early stages of software development a prototype of product, which further increases the efficiency of the developed components, is created. The disadvantages include rigid time management for software development, and the need to attract highly qualified professionals who are able to work with the necessary software tools.

In the evolutionary model, requirements for the software product can be specified gradually. The main emphasis is put on the development of the software prototype, then a complete understanding of product requirements. The advantages of this model are:

– identification of software utility at the initial stage;
– staff recruitment on demand;
– dividing of system into incremental components.

The disadvantages of this model are limited opportunities for long-term resource mobilization [3; 9].

Conclusion. Analysis the best model selection determined the evolutionary model as the most suitable one for the IEIC development. Advantages of the evolutionary model include the fact that the product is developed in the form of separate designs, but unlike the incremental model, requirements cannot be initially determined completely. In this model, specification of requirements is allowed partially and is specified with each subsequent design. Since the specificity of the development of the IEIC platform lies in the fact that at the initial stage there are no exact formulations for the platform requirements, so accordingly, there are no specific tasks for the development of the IEIC, and there is no unified design for commissioning of similar systems developed with the help of the Open source technology. In this regard based on the processes of the evolutionary model, it is necessary to work out a methodology for constructing the IEIC, taking into account the specifics of the web resource and free software products development, and for the existing technologies analysis, their functions and characteristics for developing and maintaining the IEIC.

References

1. Software Development Methodology Today. Software Development Strategies and Life-Cycle Models [Electronic source]. URL : http://www.informit.com/articles/article.aspx?p=605374&seqNum=2 (accessed October 16, 2017).

2. Inyushkina O. G. *Proektirovanie informatsionnykh sistem (na primere metodov strukturnogo sistemnogo*

analiza) [Designing information systems (using the methods of structural system analysis)]. Ekaterinburg, Fort-Dialog Iset' Publ., 2014, 240 p.

3. State Standard 15271–2002. Information technology. Guide for the application of GOST R ISO/IEC 12207 (Software life cycle process). Moscow, Standartinform Publ., 2004. 45 p.

4. STB ISO/IEC 12207–2003. Information technology. Software life cycle processes. Minsk, Gosstandart Respubliki Belorus', 2003. 52 p.

5. State Standard 12207–2010. Information technology. System and software engineering. Software life cycle process. Moscow, Standartinform Publ., 2004. 100 p.

6. State Standard 9126–93. Information technology. Software product evolution. Moscow, Standartinform Publ., 2004. 13 p.

7. STB ISO/IEC 9126–2003. Information technology. Software product evolution. Quality characteristics and guidelines for their use. Minsk, Gosstandart Respubliki Belorus', 2003. 16 p.

8. Kaner S. at al. *Testirovanie programmnogo obespecheniya. Fundamental'nye kontseptsii menedzhmenta biznes-prilozheniy* [Software testing. Fundamental concepts of business application management]. Kiev, DiaSoft Publ., 2001, 544 p.

9. Bakhtizin V. V. *Tekhnologiya razrabotki programmnogo* [Software development technology]. Minsk, BGUIR Publ., 2010, 267 p.

10. Vendrov A. M. *Proektirovanie programmnogo obespecheniya ekonomicheskikh informatsionnykh sistem* [Designing of software for economic information systems]. Moscow, Finansy i statistika Publ., 2005, 544 p.

11. Marka D. *Metodologiya strukturnogo analiza i proektirovaniya SADT* [Methodology of structural analysis and design]. Moscow, MetaTehnologiya Publ., 2003, 243 p.

12. Orlov S. *Tehnologiya razrabotki programmnogo obespecheniya* [Technology software development]. Saint Petersburg, Piter Publ., 2002, 464 p.

13. A Guide to the Project Management Body of Knowledge (PMBOK). – Upper Darby: PMI Standards Committee. 1996. Available at: http://www.softwareresearch.net/fileadmin/src/docs/teaching/SS06/PM/PMBOKINTRO.pdf (accessed 17/11/2017).

14. Bijay K. Jayaswal, Peter C. Patton. Software Development methodology today. September 22, 2006. Available at: http://www.informit.com/articles/article.aspx?p=605374&seqNum=2 (accessed 17/11/2017).

15. Kupriyanov A. V. *Tekhnologii proektirovaniya programmnykh produktov* [Software design technology]. Samara, Publishing house of the Samara State Aerospace University, 2006, 72 p.

Библиографические ссылки

1. Software Development Methodology Today. Software Development Strategies and Life-Cycle Models [Электронный ресурс]. URL: http://www.informit.com/articles/article.aspx?p=605374&seqNum=2 (дата обращения: 16.10.2017).

2. Инюшкина О. Г. Проектирование информационных систем (на примере методов структурного системного анализа) : учеб. пособие. Екатеринбург : Форт-Диалог Исеть, 2014. 240 с.

3. ГОСТР ИСО/МЭК ТО 15271–2002. Информационная технология. Руководство по применению ГОСТ Р ИСО/МЭК 12207 (Процессы жизненного цикла программных средств). М. : Изд-во стандартов, 2004. 45 с.

4. СТБ ИСО/МЭК 12207–2003. Информационные технологии. Процессы жизненного цикла программных средств. Минск : Госстандарт Республики Беларусь, 2003. 52 с.

5. ГОСТ Р ИСО/МЭК 12207–2010. Информационная технология. Системная и программная инженерия. Процессы жизненного цикла программных средств. М. : Изд-во стандартов, 2004. 100 с.

6. ГОСТ Р ИСО/МЭК 9126–93. Информационная технология. Оценка программной продукции. Характеристики качества и руководства по их применению. М. : Изд-во стандартов, 2004. 13 с.

7. СТБ ИСО/МЭК 9126–2003. Информационные технологии. Оценка программной продукции. Характеристики качества и руководства по их применению. Минск : Госстандарт Республики Белорусь, 2003. 16 с.

8. Канер С., Фолк Дж., Нгуен Енг Кек. Тестирование программного обеспечения. Фундаментальные концепции менеджмента бизнес-приложений : пер. с англ. Киев : ДиаСофт, 2001. 544 с.

9. Бахтизин В. В. Технология разработки программного обеспечения : учеб. пособие. Минск : БГУИР, 2010. 267 с.

10. Вендров А. М. Проектирование программного обеспечения экономических информационных систем : учебник. 2-е изд., перераб. и доп. М. : Финансы и статистика, 2005. 544 с.

11. Марка Д. А., МакГоуэн К. SADT. Методология структурного анализа и проектирования. М. : Мета-Технология, 2003. 243 с.

12. Орлов С. Технология разработки программного обеспечения : учебник. СПб. : Питер, 2002. 464 с.

13. A Guide to the Project Management Body of Knowledge (PMBOK) [Электронный ресурс]. Upper Darby: PMI Standards Committee. 1996. URL: http://www.softwareresearch.net/fileadmin/src/docs/teaching/SS06/PM/PMBOKINTRO.pdf (дата обращения: 17.11.2017).

14. Jayaswal B. K., Patton P. C. Software Development methodology today [Электронный ресурс]. 2006. September 22. URL: http://www.informit.com/articles/article.aspx?p=605374&seqNum=2 (дата обращения: 17.11.2017).

15. Куприянов А. В. Технологии проектирования программных продуктов : учеб. пособие. Самара : Изд-во Самар. гос. аэрокосмич. ун-та, 2006. 72 с.