Paper received: 19.04.2019. Journal of Agronomy, Technology and Engineering Management
Paper accepted: 29.07.2019. ISSN: 2620-1755 Vladisavljević and Soleša, 2019. Vol. 2 SI(1): 8-14

Review paper

THE SCOPE OPTIMIZATION OF SOFTWARE PROJECTS USING MODERN SOFTWARE TOOLS

Radovan Vladisavljević[1,*], Dragan Soleša[1]

[1] University Business Academy in Novi Sad, Faculty of Economics and Engineering Management in Novi Sad, Cvećarska 2, 21000 Novi Sad, Serbia

*Corresponding author:
E-mail address: radovan.vladisavljevic@fimek.edu.rs

ABSTRACT: The paper aims to highlight software project management tools and techniques that aim to optimize project scope. Project management is one of the most sensitive management techniques that aim to deliver the highest quality result possible. Throughout the work, we will show the impact of modern software tools that, in addition to optimizing the scope of the project, increase the quality of not only the results of work but also the management of project tasks.

Key words: *software project management, project scope, software tools, software project management techniques*

INTRODUCTION

Modern business requires the development of new solutions that will solve problems that the business has not encountered before. On the other hand, software becomes a strategic resource of the company, without software support, many business processes cannot take place. For this reason, software production is becoming an extremely important branch of the industry. An engineering approach that for decades, if not centuries, has been adequate in the domain of finding solutions has proved inadequate in the domain of software engineering. In other words, the software is a particular branch of engineering for which not enough tools and techniques have been developed. All of this led to the creation of inadequate software products. With the development of information science and software engineering, new software development models are emerging as well as tools that improve the process of creating a new software product. Namely, assistive tools can largely replace some, but by no means all, of the deficiencies of inadequate software development models.

DEFINITION OF PROJECTS AND PECULIARITIES OF SOFTWARE PROJECTS

By definition, a project is a temporary endeavor that aims to create a unique service, product or result (PMBOK, 2017). What should be emphasized is that the result of a project is often not of a temporary nature, which directly affects quality. Otherwise, the concept of the project itself is not new, but managing a project is often not effective enough. To justify the use of a project approach, the problem itself must be sufficiently

complex and relatively well structured. Otherwise, the project approach is not adequate due to its nature. Numerous other management tools are far better choices for trivial and / or under-defined problems.

On the other hand, we have a particular application of the project approach in the software industry. Software is by its very nature an intangible product that is heavily influenced by the customer. The first step in creating software is to collect customer requests, unfortunately, requirements change frequently and require constant review (Wiredu, 2020). In such conditions, the classical engineering approach is not adequate. That is why new project management models have evolved that are far more flexible and easier to manage.

Project management has always been planning-based (Pica, 2015). This is also true of software projects; however, the time spent planning in software development must be rigorously optimized. A mitigating factor in software projects is the emergence of software tools that can greatly facilitate the management and planning process.

Tool-driven software development is the new paradigm behind many new software product creation models. Tools for group collaboration lie at the heart of software project domain efforts (Wiredu, 2020). These tools connect team members and provide opportunities for contracting clients to engage in the early stages of planning.

However, when the software system under construction becomes too complex, it tends to exclude the human factor either through a different organization of work or through tools (Shishkov, 2020). At first glance, this setting goes beyond the basic goals of software projects, but logic dictates that the human factor must be relativized at some point. This is especially important for systems that go beyond the understanding of the individual. Tools become an integral part of software project management without which it is difficult to imagine the successful completion of project tasks. In each segment of the software project, specialized tools can be found to facilitate the work on the software project.

SCOPE OF PROJECTS

The scope of the project depends on three key elements: quality to be delivered, time to be spent and resources to be engaged. These three key elements are the basic pillars of project management. Project quality can be defined as a series of activities that must be carried out for a project to meet agreed specifications. Specifically, the keyword here is an activity for the simple reason that quality control enters the activity. When planning, activities that are part of the quality assurance system should be envisaged. If these activities are neglected, the quality may be unsatisfactory.

The time it takes to complete project activities is defined in the planning process. However, it is in this segment that there can be gross omissions. In such a case, the recovery of the project can be problematic.

Resources are not just about money; this segment is far broader and includes other hard-to-define moments in addition to financial moments. Knowledge and experience are typical project resources that are difficult to define. They are abstract in themselves, but the members of the project team in charge of carrying out the project are the bearers of knowledge and experience.

All told so far applies to software projects, especially in the area of human resource management. The software is mainly a product of the experience and knowledge of the project participants. On the other hand, when the software becomes too complex then the division of work can take place.

PROJECT LIFE CYCLE

In practice, we can usually identify the following stages of the project life cycle: project initialization, planning, execution, control, and closure. It is about starting the project, organizing and preparing, carrying out the work and closing the project (Montanari, 2015). The purpose of this division is that in the way it is easier to control the flow of the project, it must be remembered that the project is essentially a tool to solve complex problems. From this, it follows that a successful framework requires a clear framework in which to work. There is a strong correlation between business architecture and project scope. This relationship fluctuates throughout the project life cycle (Davis and Radford, 2014).

Business architecture is a term that in many cases refers to the process by which business is performed in companies. The difference between architecture and process is very small and in this paper, these two terms will be used with the same meaning. On the other hand, it must be emphasized that it is not uncommon for a business architecture change to occur at the beginning of the project life cycle.

The sooner changes are identified and incorporated into the project, the better and easier the solution. No major costs are expected in the planning process and the first steps of the life cycle, the amount of work done is relatively modest. However, the more the project progresses, the more expensive and complicated the changes to implement.

For this reason, it seeks to introduce a range of organizational and technical solutions to involve the contracting authority at all stages of the project life cycle. The software does not have its physical representation, which is why it is difficult for contracting authorities, who often lack technical experience, to understand how complicated and expensive the changes can be.

Life-cycle theory can be applied to other segments of computer science, for example, data engineering can also be analyzed using life-cycle stages. In this way, it is possible to develop a unique methodology (Shan et al., 2017).

TOOLS IN INDIVIDUAL SEGMENTS OF THE SOFTWARE PROJECT LIFECYCLE

Lifecycle analysis and linking to tools are critical to successfully managing software project efforts. Practically every step of the life cycle offers new opportunities but also challenges. What should not be forgotten is that the closer the project is to completion, the fewer opportunities to improve performance. For this reason, the tendency is to restore control of the project as early as possible (Dimitrov, 2020).

The earlier the project, the easier it is to plan and involve external stakeholders. In the later stages, when the project becomes complex, careful management of the information is required. The initial stages of the project life cycle are mainly planning and communication-oriented.

The first phase of the life cycle is the initiation of the project, which is most often initiated by an external factor. It is the contracting authority or entities of the software project, some variations include internal requirements for the new project. However, most software development methodologies are outsourced and easier to treat internally.

This stage is dominated by tools that are based on improving communication. Various visual tools and rough plans give a general picture that shows the purpose and nature of the future product of the project effort. Tools like the mind map can help with the first segments because they give a graphically clear and intuitive representation of what one wants to achieve.

Likely, the first steps of the project effort do not address specific technologies and solutions implementation. The initialization of the project gives a rough outline of the scope and resources needed to complete the project. In this part, contracting authorities have an extremely large influence.

Requirement engineering is the first step leading to the next step, here the data required to move to the planning segment is collected. Scenario method, observation, and documentation analysis are some of the most common tools at this stage. A script method is a powerful tool that is intuitive and relatively easy to use. System researchers task respondents for whom a software product is created to describe all the jobs and procedures they use in a given process. The power of this tool lies in the fact that respondents can easily identify with a given situation and more easily communicate with examiners.

The observation method has several segments, one of which is the method based on questionnaires and interviews. Another approach is to participate directly in the work process, here the respondents observe how a process takes place. This method has a large knowledge base from which ready-made questionnaires or procedures can be drawn. The processing of such results is relatively simple and is based on the detection of regularities in mass phenomena.

Document analysis is the next tool to discover how the process is going in the organization under review. This method is historical and analyzes the previous states of the process; however, this method has a strong objective character so that it is less susceptible to misinterpretation.

In the higher stages of the project life cycle, the focus changes from a rough picture to detailed plans. Namely, in the initial stages when a project is initiated, much important information remains hidden; with the exploration of the system comes several important details. These details give a clearer picture of what needs to be done to complete the software successfully.

The planning process inputs are the results of research that can be used to divide work and create plans that further develop a software project. The basic tools that appear in the planning process are work breakdown structures (WBS), critical path method (CPM), Gantt chart, PERT analysis, etc.

WBS gives a clear picture of what needs to be done as well as a rough outline of the sequence of activities. Every project, even a software project, is, in fact, a series of activities that must be undertaken to produce the desired result. WBS is a tool by which

activities can be clearly and unequivocally shared, this is the first step towards forming development teams. Job sharing is a basic step towards other tools.

CPM is a powerful method used to analyze priority jobs. The logic is that one project lasts as long as the longest series of activities. Any activity that is on a critical path can affect the length of the entire project. The purpose of CPM is to protect the integrity of the project by identifying individual activities. Activities not on the critical path have a slightly smaller impact on the duration of the project. The danger is that activities that are not on the critical path are ignored, in such cases, the priorities are changed and a new critical path is formed.

Gathogram is a special tool that graphically displays activities, their time duration, and interdependencies between activities. This tool is extremely useful not only for planning but also for controlling progress.

PERT analysis, this tool uses several techniques that are part of network planning. Unlike the Gantt chart, a network plan has a focus on interdependencies. In the Gantt chart, we can see activities and duration, but the interdependencies are not well represented. On the other hand, the network plan gives a far richer overview of interdependence. PERT analysis provides a mathematical model that can analyze the critical path as well as the possibility of applying the scenario. PERT analysis scenarios provide an overview of the best and worst possible outcomes.

The phase associated with the execution of the software project aims to implement the plans. Many of the tools that appear in the planning phase are also used at this stage. The purpose of a group of performance tools can hardly be separated from control. Dominated by reporting tools, this toolkit provides a cross-section of project development progress.

Many project management software tools have built-in reporting generators to create detailed reports. The use of artificial intelligence and expert systems greatly influence the quality of reports.

Closing a project is often the least represented phase of the life cycle. However, it is important not so much for the current project but future projects. At this stage, there must be a consensus that the project is completed, and the technical and legal documentation must be finalized here. The financial structures at this stage must be completely closed.

One may ask whether it is possible to successfully run software projects without the use of these tools, or rather, whether it is possible to neglect software that helps create a better, better quality project. The answer is, of course, that it is not possible or very difficult to do.

What we need to emphasize is that all the tools that we have listed in this paper can be integrated into a specific software product. Software applications like MS Project have integrated project management tools. In recent years, Web applications have emerged that, in addition to these tools, provide greater integration of team members and better control of project work.

Codesharing can be done using repositories like the GitHub service. Thus, even at the activity level, the improvements generated using tools can be observed. Web

applications like JIRA can directly track software development; a similar tool is YouTrack, which offers the ability to work in groups.

Today, new methodologies are being developed that integrate tools and organizational techniques for managing software projects. One of the more recent methodologies is Manutelligence, which aims to integrate the best project development methods (Cattaneo et al., 2019). This platform is used for numerous projects that may not necessarily be in the domain of software product development. Also, there is The Manutelligence consortium that aims to bring together highly qualified cocoa staff to contribute to project development.

The motives that drive organizations to use new project management tools and techniques are: Compression of the Product Life Cycle, Knowledge Explosion, Triple Bottom Line (planet, people, profit), Corporate Downsizing, Increased Customer Focus, and Small Projects Represent Big Problems (Larson and Gray, 2013).

CONCLUSIONS

Tools can greatly shorten and improve the development of a software project, what should not be forgotten is that humans are an important segment without which tools would be pointless. However, collaborative work is of great importance for the successful completion of software projects. Modern methodologies that belong to the agile development method family rely on the tools and techniques we have presented in this paper.

Of course, this is a very small oeuvre of the tools that we could present in this paper, it must also be emphasized that new techniques and tools are being introduced every day, aimed at managing software projects. The reason for this is that many project activities can be shortened or run in parallel without compromising project quality.

Tools and techniques cannot compensate for poor project management; the project team led by the project manager must have all the necessary knowledge and experience to be able to use the tools to the full.

REFERENCES

CATTANEO, L., CASSINA, J., PETRUCCIANI, M., TERZI, S. and WELLSANDT, S. (2019). Models, Methods and Tools for Product Service Design: The Manutelligence Project. *Springer, Milan, Italy.*

DAVIS, B. and RADFORD, D. (2014). Going Beyond the Waterfall Managing Scope Effectively Across the Project Life Cycle. *J. Ross Publishing, Pine Island, USA.*

DIMITROV, D. (2020). Software Project Estimation: Intelligent Forecasting, Project Control, and Client Relationship Management. *Apress: Toronto, Canada.*

LARSON, W.E., and GRAY, F.C. (2013). Project Management: The Managerial Process. *The McGraw, New York, USA.*

MONTANARI, F. (2015). Projects and Their Life Cycles: Some Current Views in Project Life Cycle Economics (Ed. Pica, M.). *Gower, Farnham, UK.*

PICA, M. (2015). Project Management, Projects and Their Life Cycles: Historical Views in Project Life Cycle Economics (Ed. Pica, M.). *Gower, Farnham, UK.*

PMBOK® (2017). Guide A Guide to the Project Management Body of Knowledge. Six edition. *Project Management Institute, Pennsylvania, USA.*

SHAH, M.S., WELCH, J., DAVIES, J. and GIBBONS, J. (2017). Software Project Management for Combined Software and Data Engineering in Software Project Management for Distributed Computing: Life-Cycle Methods for Developing Scalable and Reliable Tools (Ed. Mahmood, Z.). *Springer, Swindon, UK.*

SSHISKOV, B. (2020). Designing Enterprise Information Systems: Merging Enterprise Modeling and Software Specification. *Springer, Sofia, Bulgaria.*

WIREDU, O.G. (2020). Global Software Engineering: Virtualization and Coordination. *CRC Press, Boca Raton, USA.*