The Eurasia Proceedings of Science, Technology, Engineering & Mathematics (EPSTEM)
ISSN: 2602-3199

The Eurasia Proceedings of Science, Technology, Engineering & Mathematics (EPSTEM), 2024

Volume 28, Pages 533-553

ICBASET 2024: International Conference on Basic Sciences, Engineering and Technology

Software Project Management and Their Economic Evaluation in terms of Enterprise Sustainability

Ibrahim Krasniqi
University Haxhi Zeka

Naim Ismajli
AAB College

Ganimete Krasniqi
Ministry of Environment, Spatial Planning and Infrastructure

Abstract: Nowadays, we are witnessing that projects, computers and software as an integral part of them infiltrates or rather ruthlessly penetrates all spheres of business and daily life of a modern society and directs enterprise development, efficiency, productivity, competitiveness, image and ultimately in a way their very existence. In recent decades, the software industry and its products have become one of the factors without which the normal operation of contemporary economies cannot be imagined, however, software development often presents almost insurmountable difficulties.The research is related exactly to the analysis of software projects and their impact in the sustainability of enterprises. It is difficult to imagine any enterprise that claims to be competitive, to prepare, realize and monitor their projects without software support, be it standard software or specially developed for the needs of the enterprise, where software development comes into play. During the research combined methods are utilized. The methodology used includes the theoretical analysis which is based on the current literature from this field, followed by "case studies" that deal with the development of software projects in Kosovo by conducting interviews with their IT managers using face to face and semi structured interviews and finally based on gathered data the development of a concrete software project has been simulated. Finally the outcome from research is that always software development has to be identified with the project objectives of the project contractors and aims to meet their requirements and expectations. Software project team's work for years in the development of complex and professional software projects, where competence in the relevant field as well as motivation play a key role in the development of successful business projects.

Keywords: Project management, Software projects, Business engineering

Introduction

In the modern business environment, software projects play a pivotal role in the operations and success of enterprises across various industries. Effective software project management is essential for ensuring the timely delivery of high-quality products and services, meeting customer expectations, and maintaining competitive advantage (Aarseth et al., 2017). However, alongside the technical aspects of project management, there is an increasing recognition of the importance of economic evaluation in terms of enterprise sustainability (Acar, 2017). Enterprise sustainability encompasses the long-term viability and resilience of organizations in economic, social, and environmental dimensions. It goes beyond short-term profitability and considers the broader impact of business activities on stakeholders, society, and the planet (Albertao et al., 2010). In this context, evaluating software projects from an economic standpoint becomes crucial not only for financial performance but also for sustainable development. This study aims to explore the intersection of software

project management and economic evaluation within the framework of enterprise sustainability (Alharthi. et al., 2016). By examining the economic aspects of software projects, including costs, benefits, risks, and returns on investment, we seek to understand their implications for the overall sustainability of enterprises. Through rigorous analysis and empirical evidence, we aim to shed light on how organizations can optimize their software project management practices to enhance sustainability outcomes (Becker et al., 2017). The significance of this research lies in its potential to inform decision-making processes within organizations, guiding resource allocation, investment strategies, and project prioritization. By integrating economic evaluation into software project management practices, enterprises can not only achieve short-term objectives but also contribute to long-term sustainability goals (Beghoura et al., 2017). Moreover, by considering sustainability criteria in project decision-making, organizations can align their activities with societal values, regulatory requirements, and environmental stewardship *Calero and Piattini (2017). In the subsequent sections of this paper, we will delve into the theoretical foundations of software project management and economic evaluation, review relevant literature, propose a conceptual framework for analysis, and present empirical findings from case studies or data analysis (Calero et al., 2013). Ultimately, we aim to provide insights and recommendations that can support practitioners, policymakers, and scholars in advancing the sustainability agenda within the realm of software project management (Garcıa-Mireles et al., 2018).

Indeed, we are witnessing that computers and software as an integral part of them infiltrates or rather ruthlessly penetrates all spheres of life of a modern society and directs development, efficiency, productivity, competitiveness, image and ultimately in a way their very existence (Garcıa-Mireles et al., 2018). In recent decades, the software industry and its products have become one of the factors without which the normal operation of contemporary economies cannot be imagined, however, software development often presents almost insurmountable difficulties (Huemann, & Silvius, 2018). It is difficult to imagine any enterprise that claims to be competitive, without software support, whether it is standard software or specially developed for the needs of the enterprise, where software development comes into play. Software development is identified with the project objectives of the project contractors and aims to meet their requirements and expectations (Kern et al., 2015). Software project teams work for years in the development of complex and professional software projects, where competence in the relevant field as well as motivation play a key role in the development of successful business projects (Kern et al., 2013)

Statement of the Problem

The main objective of this study has been to analyze the problem of identification, planning and evaluation of software projects to give a modest theoretical contribution regarding the methods and their specifics. The research in this study is based on the fact that:

- The development of software projects, as well as many investments from the field of information technology, presents a high degree of risk, which is especially related to the change in user preferences and their unrealistic expectations. All this is related to the very nature of the software. It is a very specific, immaterial product and as such cannot be seen or touched. This makes the development and management of software projects much more complex than the development and management of classic projects. In fact, the software project manager can monitor and control the development of the project only based on a solid documentation of it. The chances that this determination / assessment of software project development is not objective are very high.
- The goal of software project management is to increase productivity, raise quality, and minimize costs during software creation. This requires the coordination of influential factors, the balance between which enables the development of the software project within the anticipated budget, anticipated time limits and planned expenses. In this paper, attention is focused on the identification and evaluation of software projects. Other phases in the project management cycle have been addressed as necessary.

Research Questions

1. What is the difference between other projects and software projects?
2. What is the most suitable method for evaluating the software project?
3. In Kosovo, which method is more efficient to use?

Literature Review

Sustainability is a multifaceted term; it has different meanings to different groups and contexts. A meaningful and reasonable understanding of sustainability implies collective efforts of socially responsible actions and their impact on others especially in the future. Huemann and Silvius, (2017), defined business sustainability as "meeting needs of the firm's direct and indirect stakeholders, such as shareholders, employees, clients, pressure groups, communities, without compromising its ability to meet future stakeholder needs as well" . Some research considered when organizations utilize recent IT/IS in favour of enhanced performance levels that last for longer time, is called sustainable IT. In this regard, information systems play an important role in improving the efficiency of firms' operations and supply chains, which links to sustainability (Lago et al., 2018). Researchers Marnewick, C. (2017), believe that sustainable IT is characterized by the application of IT practices and technologies for the benefit of different stakeholders to ensure long-term benefits in economic, social, and environmental sustainability pillars (Martens, M.L. and Carvalho, M.M. 2016), Previously, sustainability has been envisaged as a constraint in the business domain (Økland, A. 2015). The success of business organizations have been founded upon minimizing costs and increasing revenues. Software project management and its economic evaluation in terms of enterprise sustainability nowadays is becoming very popular and important in general. In order to understand the topic and aim the research will analyze the terms and bescribe their role in enterprice sustainalbility.

Software Project Management (SPM) Frameworks: Numerous SPM frameworks exist, such as Agile, Waterfall, and DevOps (Pohludka et al., 2018). These frameworks offer different methodologies for managing software development projects, each with its own set of advantages and challenges. Enterprise Sustainability: Enterprise sustainability encompasses the long-term viability and resilience of an organization in terms of economic, environmental, and social factors. In the context of software project management, sustainability involves ensuring that software projects are delivered efficiently, effectively, and in a manner that aligns with the organization's overall sustainability goals (Savitz. 2013).

Economic Evaluation of Software Projects: Economic evaluation techniques, such as Cost-Benefit Analysis (CBA) and Return on Investment (ROI), are commonly used to assess the financial viability of software projects (Silvius et al., 2018). These evaluations consider factors such as development costs, potential revenue or cost savings, and the overall economic impact on the organization. Alignment with Sustainability Goals: Researchers have highlighted the importance of aligning software project management practices with broader sustainability goals. This alignment involves considering factors such as resource efficiency, waste reduction, and the environmental impact of software development processes (Sumner, 2018).

Sustainable Software Development Practices: Adopting sustainable software development practices, such as green computing and energy-efficient programming techniques, can contribute to both economic savings and environmental sustainability (Statista.com. 2021). These practices aim to minimize the carbon footprint and ecological impact of software projects while optimizing resource utilization. Challenges and Opportunities: Despite the growing recognition of the importance of sustainability in software project management, organizations face various challenges in implementing sustainable practices (Turner et al., 2013). These challenges include balancing short-term economic concerns with long-term sustainability goals, overcoming resistance to change, and integrating sustainability considerations into existing project management processes (Yunis et al., 2013).

Future Directions: Future research in this area may focus on developing comprehensive frameworks and methodologies for integrating sustainability into software project management practices. Additionally, there is a need for empirical studies to assess the real-world impact of sustainable software development practices on enterprise sustainability metrics (Zerbino et al., 2021). This literature review provides an overview of the current state of research on software project management and its economic evaluation in the context of enterprise sustainability (Ziemba, 2019), It highlights the importance of aligning software project management practices with broader sustainability goals and identifies opportunities for future research and practice in this area.

Software Projects, Their Types and Features

It is difficult to imagine any enterprise that claims to be competitive, without software support, whether it is standard software or developed specifically for the needs of the enterprise. The software project represents that type of project where mainly or exclusively software is developed. However, it should be borne in mind that the types of projects dealing with software development are very different. According to Ziemba, E. (2019), software projects are of different types such as:

- development projects,
- prototype projects,
- evolutionary projects,
- migration projects,
- maintenance projects,
- projects for integration, and
- projects for installation.

There are major differences between the Software and other projects such as:

- software is immaterial, not physical,
- software is not produced, but developed,
- the software does not wear out,
- the development and delivery of the software, most often, is contracted as a whole and not as parts (unless it is developed with separate modules).

The software presents a summary of ideas, designs, instructions and formulas. Next, let's see what the project represents and what are the differences between software projects (Sumner, M. 2018). In addition to what was said above, there are also some other features that characterize software projects, such as:

- In most cases, the realization of an idea is done for the first time,
- Long duration,
- Great opportunities for presenting problems, respectively different risks.

Acordıng to Schieg, M. (2009), Features of the software projects are:

- Intangible product,
- It is not possible to accurately measure the development time of the project,
- Software development does not flow in a defined manner,
- The development process is not yet clearly defined,
- Distribution of tasks,
- Software engineering is not a natural science,
- High level of abstraction and at the same time low level of standardization.

In the literature, we will also come across a classification based on the size of the project, such as the size of the software project, depending on the hours committed to its development:

- small 1 – 250 hours
- medium 251 – 5,000 hours
- great over 5 000 hours

According to the application and the groups to which they are dedicated:

- Individual software (individual production, contract production)
- Standard software (standardized solutions, user groups)

Another division of software projects is the classification of software according to the degree of difficulty - its complexity; classification that when put in relation to the so-called maturity level of the process model (of software development) Capability Maturity Model - CMM, a very clear overview of the company's competitive capabilities in the software market will be obtained, and also indirectly reflects the state of development of the software industry.

Capability Maturity Model - CMM can be said to represent the level of experience achieved by the enterprise in the development of software projects, according to which there are five degrees of maturity of the development process, respectively of the enterprise's ability to develop software projects. The aforementioned division or classification is as follows:

Type "S" software can be fully described through a formal specification, which means that the generated software corresponds to the specification faithfully. Typical examples of "S" (Specification) software are programs for sorting data, or for calculating various mathematical functions.

"P" (Problem) type software represents that software that solves a defined and specific problem, while typical examples of "P" software represent programs that deal with different calculation models from the field of constructions (e.g. e.g. statics) or meteorological forecasts.

"E" type software (Embedded) represents such software or application that is embedded - Embedded (software installed on various devices). As we can see, "S" type software compared to "P" and especially "E" type software appears to be easier to develop and is more stable in terms of evolution.

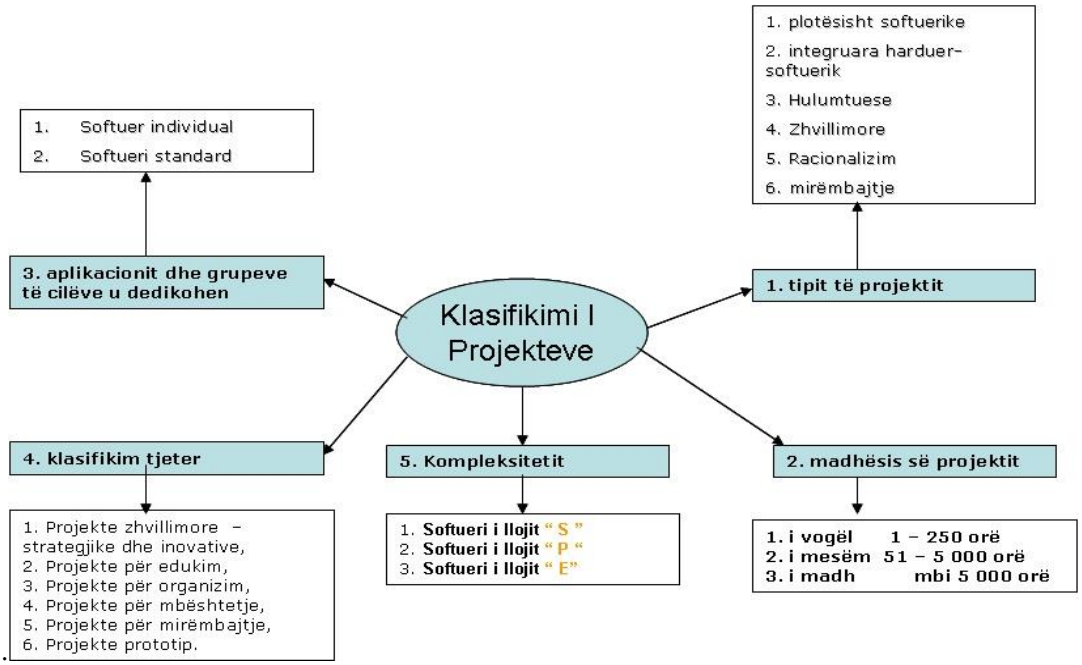

Figure 1. Classification of software projects

Software Project as an Investment

Ideas for investment projects can come from both the state and the private sector, while in terms of software projects, which can also apply to any other type of project, before the realization of these ideas it must be decided whether, in really, was the project in question carried out or not? In this part of the paper, we will examine only one of the three aspects of decision-making, namely the decision to implement the project, that is, we will examine only the types of expenses and the types of benefits related to software projects.

As a measure for the measurement of personnel expenses, it is presented M/NY month/person, while for the volume (size) of the product (product scope) it is Lines of Code - the number of lines without counting comments as well as Function Points - the number of functions separately . In various literature, we will come across the division of expenses, such as in personnel expenses, material expenses, expenses for foreign services, expenses for taxes and various taxes, etc., as well as capital expenses.

Software Market and Industry

The software industry is the collective name for the business that deals with the development and maintenance of software (Økland, 2015). This industry also includes activities, namely services that are related to software such as: training, consulting and maintenance. Market segmentation represents an essential step in the marketing planning process, because the main purpose of market segmentation is to enable the company to channel its commitments in those areas where the opportunities are greater and to find ways to differentiate its from the competition. As the first division of the software market, the so-called division into the primary and secondary branch (branch) is presented.

The primary branch includes those enterprises that specialize in software development or services in the market, related to both software and hardware (giving advice), while the secondary branch includes those enterprises that self-developed or software related services they offer within the company, while they do not offer either

software or software related services in the market. In fact, the secondary sector is considered to belong to all sectors of the economies of developed countries, because it is difficult to find any sector where software is not found as part of various products and services, such as:

- telecommunication,
- electrical engineering,
- automotive industry,
- pharmaceutical industry,
- financial services (banks, insurance companies),
- traffic, etc.

Another division of the overall software market is that of horizontal and vertical software products (Martens, M.L. and Carvalho, M.M. 2016). The vertical market deals specifically with specialized industries, while the horizontal market tends to serve a much broader and more general user base.

Table 1. Software project market division

Horizontal software	Vertical software
general calculation software	software for complex calculations (various enterprises)
word processors	software for managing different fuel pumps
project planning software	oil exploration management software
IP(Internet Protocol) network monitoring software	geological telemetry monitoring software

Success of Software Projects

Successful software development largely depends on the quality of software project management. The essential goals of software project management are related to increasing productivity, ensuring a defined quality, as well as reducing costs for achieving the objective during the development of the software project (Huemann, M. and Silvius, G. 2017).

According to Marnewick, C. (2017), three primary software project management strategies are distinguished:

- maximizing the satisfaction (fulfillment of requirements) of the users,
- minimizing costs and time for software development, and
- minimization of errors (bugs).

The management of software projects is different from the management of other projects, for the reasons mentioned earlier in this paper, and in the following we will focus in more detail on the factors that determine the level of success of software projects. There are numerous statistical reports related to the fate of software projects that contain the answer to the question "what happened to software projects?" The results were depressing.

- Delivered but not working 47%
- Paid but not used 29%
- Repaired, but not put into use 19%
- Repaired and put into use 3%
- 2% were put into use without major problems

The reasons for software project failures according to different authors are different, but in this paper we will present only a few that are mentioned by most authors.

- unclear requirements or inefficient planning,
- lack of process models or inadequate management information system,
- inefficient system of monitoring, evaluation and control,
- prices and unrealistic planning of objectives,
- insufficient communication and excessive documentation orientation,
- insufficient quality assurance as well as non-definition of "core functionality"

and the factors that determine the fate of software projects, according to the Standish Group, arE:

- user participation 15.9%
- implementation support 13.9%
- clear formulation of requirements 13.0%
- adequate planning 9.6%
- real expectation 8.2%
- finer project milestones 7.7%
- competent personnel (staff) 7.2%
- ownership 5.3%
- clear vision and objectives 2.9%
- intensively engaged personnel 2.4%

Management of Software Projects

The objective of software project management is to develop software on time, with quality, and at an acceptable budget, software that works, that is developed on time and on budget, and that can be maintained and reused. To achieve these goals, successful management is needed in the development of software projects. So, in order not to risk developing a low-quality software product, i.e. a product that will not meet the expectations and will not respond to the contractor's requirements, in order not to risk being delayed in the realization of the project, in order not to exceed the foreseen budget, in order not to engage unmotivated personnel (or to risk that they become demotivated) both in number and inadequate qualifications, and that all of this ends up with entropy of the so-called software project system, it is necessary to use the methods offered by the discipline of modern project management. So, with project management we coordinate the project's objectives, work and resources to achieve the goal we aim for, always accompanied by different definitions (in people, resources, time, etc.), while in terms of software project management, time of programming as "fun" that the weather has passed, while software development in a professional way is always, or almost always, accompanied by a lack of different definitions (time, tools, etc.). The generation of software until today has been managed to dominate only partially, and as evidence for this we have the various reports such as much quoted standish group report, with statistics for different years. The reasons for which project management is needed can best be seen from the so-called magic square, presented in Figure 4, where in order to achieve the appropriate cost of the project, quality, quantity, and duration are presented as influential factors. and expenses. The main problems arise from the task that project management has to balance between these factors, for the reason that quality, quantity and productivity must be increased, while expenses and duration must be reduced, respectively shortened. But in order to better understand the square in question, let's see the importance of the mentioned factors for the evaluation and management of software projects. Quality - refers to those characteristics of the software which will enable its use, respectively its operation in the required or suitable way. Quantity – is treated as a very important attribute or characteristic, which is expressed in measurement units such as Line Of Code - LOC, etc. which is used the most, but which, as we said, is not the only one, while the expenses caused by the same are often presented in Person Month- PM (Calero & Piattini, 2017).

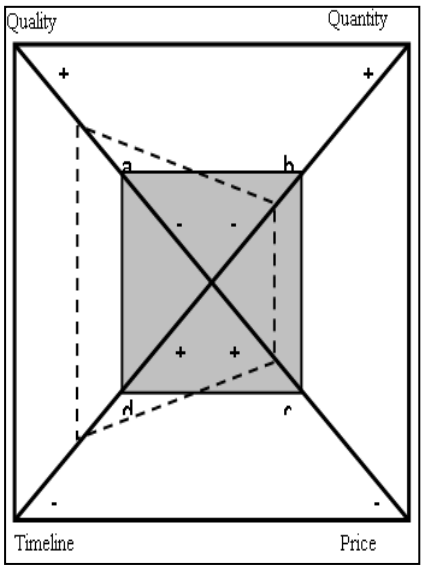

FigurE 2. Magic square

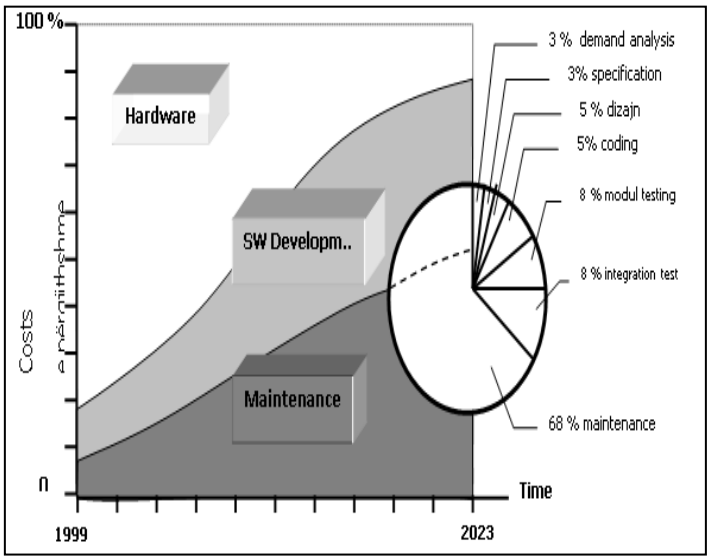

Figure 3. Software investment trend

The duration of the project - also represents a very important factor in the management of projects, especially software ones; for example, the correlation or relationship between the costs and the duration of the project is not linear as it might seem in advance, because in fact the costs will increase with the reduction or shortening of the duration of the Project

Software Project Management Process

Project processes can generally be divided into two main categories, namely:

- project management processes, and
- product-oriented processes.

Project management processes are concerned with the description and organization of work within the project, while product-oriented processes are concerned with the specification and creation of project products. Product-oriented processes are usually defined according to the Project life-cycle and are distinguished according to the field of application and consists of five phases of project management that we will mention below (Anaya, 2019).

- The planning phase of the project contains the creation and protection of the realized scheme for the fulfillment of the purpose and objective of the project.
- The implementation phase includes: training and management of the project team, realization of the project plan, recording of actual resources for use, modification of the project plan, collection and distribution of project information in progress, managerial relations and dispute resolution.
- The control phase of the project should ensure the continuation of the project according to the plans and the realization of the decisive goals.
- Project closure involves the formal acceptance of project results and includes the following activities: post-project status review, summary report preparation, project data archiving, project inventory sale, and project team distribution.
- Project audit and its commissioning.

Software Project Life Cycle (Software Life - Cycle)

The life cycle of system development represents the process of understanding how an information system can support business needs, designing the system, constructing it and delivering it to users Acar, H. (2017), In the standard IEEE Glossary of Software Engeering Terminology, Software Life Cycle presents: "The period of time that begins when the software product is conceived and ends when the product is no longer available for use (Anaya, 2019). A typical software life cycle contains: the requirements phase, the design phase, the implementation phase, the testing phase, the installation, testing and verification phase, the operation and maintenance phase, and sometimes the superstructure phase." To create such a software component, the following activities are carried out:

- task analysis, specification of requirements as well as documentation and control of requirements,
- conception of the solution, documentation and control of the concept,
- detailed design of the solution, documentation and control of the design,
- coding and testing that part of the program developed for the respective component,
- integration with parts of programs (codes) of other components (separately),
- installation and testing of the component as part of the overall system.

Definition of Goal and Objectives in Software Projects

Project objectives are derived from the project goal, which represents the highest level orientation point. They should describe the volume of the project, its breadth and specific goals (Sumner, M. 2018). The objectives and goal of the project must, before the start of the project, be clearly defined and there is full agreement about them, because the outputs of the project will be developed on the basis of these objectives and goals and not the other way around. The objectives of the software project are subject to the SMART framework, which means that when setting/defining the goal, care should always be taken that it - the goal fulfills some criteria that actually

make up the acronym SMART, (Specific - specific, Measurable - that can to be measured; Achievable, Realistic, Time-related).

Software Project Planning

In planning the development of software projects in the aforementioned source, three levels of the process are distinguished, including:

• Process architecture – shows how the software development flow should be specified, what standard process elements exist (should be taken into account) and how the mutual influence between them should be described,
• Process model – to establish the course of action (procedure) for the development of a software product.
• Project plan – a project plan is formed for each concrete software development. This plan is therefore formed by the project manager and is oriented towards one or more process models (incarnations of a Process Model).
Further along the process, software project development planning adopts the following activities (Marnewick, C. 2017). The basis for planning all activities is the WBS, or project hierarchy structure, which makes the hierarchical division of the project into clearly defined tasks or activities, thus identifying the work that must be done to complete the project. The structure of the project plan can be in the form of a table or graph.

Table 2. Software project development planning

Project definition	Evaluation	Risk evaluation
Definition of objectives and requirements.	Determination of product dimensions.	Risk analysis Categorization of risks
Selection of the development process.	Time planning (scheduling).	Risk assessment
Definition of work.	Cost estimation and budgeting.	Preparation of risk response plans

Planning Procedures

Planning and follow-up of the plan represent the most important tasks of project management, while quality control and assurance are closely related to them. In fact, a plan requires answers to these questions: what, how, how much (expenses), who, when and with what, while in planning procedures we have to answer the question how?

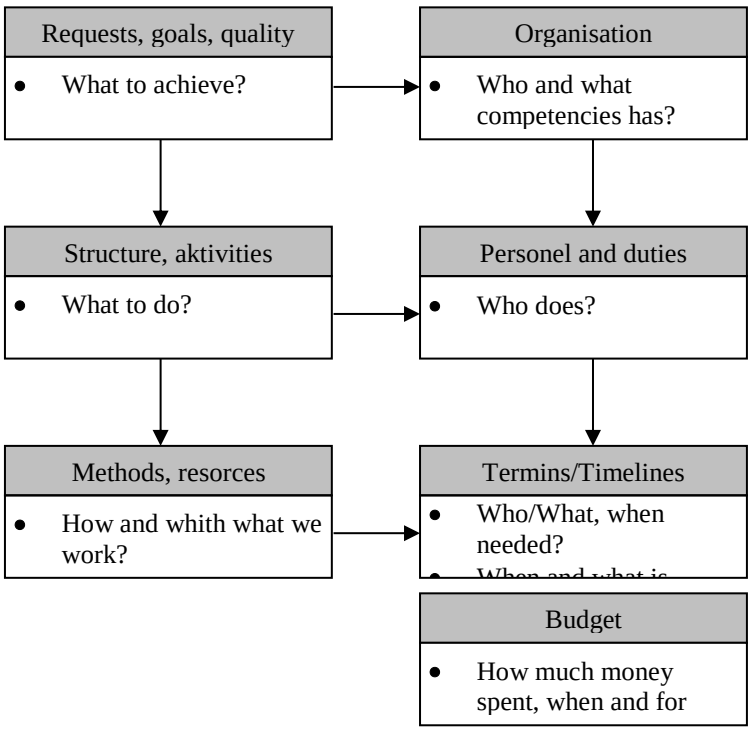

Figure 4. Plan flow formed by interdependence of plan parts

Process Models of Software Projects

Process models present concepts on which projects are based, and any software development must be done within defined organizational frameworks, whereas a process model describes such a framework.

Table 3. Software project process models

According to Balzert a defined process should define the following:	List of process models for software projects:
• sequence of work flow, • each activity that must be carried out separately, • defining the parts of the product, • product performance criteria (when it is a completed product), • necessary qualifications of collaborators, • responsibilities and powers, • applicable standards, directives, methods and tools.	• Code And Fix • Classic "Waterfall" model • "Spiral" model • "V" model • Evolutionary / Incremental Model

Evaluation of Software Projects

Types of Procedures

The evaluation of software projects represents an attempt to estimate the planned expenses for the development and maintenance of software projects, which is done through the analysis, determination and evaluation of factors influencing the software project, as a basis for evaluation (as inputs - input data) and also also in terms of the form of presentation of outputs (results), e.g. Line of Code, Function Point, Object Point, etc.).

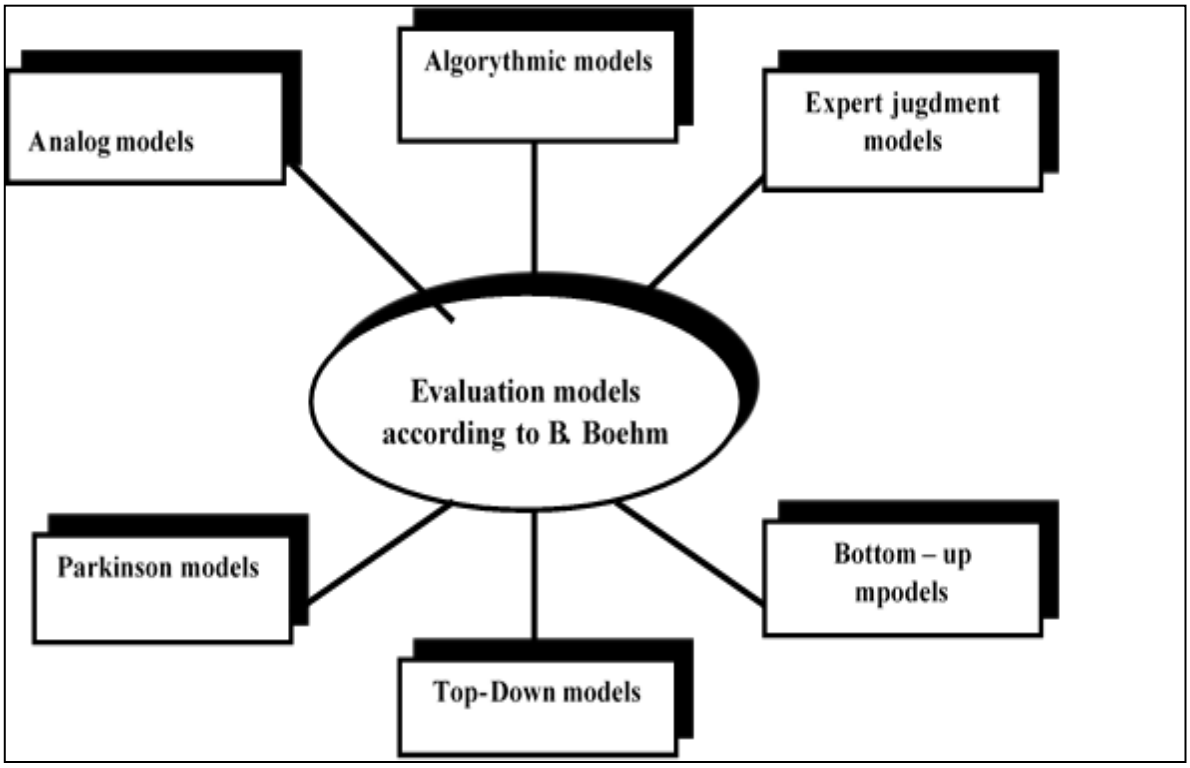

Figure 5. Project evaluation models

However, when it comes to the costs of a software project, the factors with influence in determining the costs of the development of the software project (cost-factors) should definitely be mentioned, such as Factors (performance) of the product, of the computer, of the personnel and of the project that we will explain in more detail when we talk about the Constructive Cost Model - Co Co Mo methods (constructive cost model, Funnctin Poin method and Object Point method. Methods of evaluation of software projects according to (Martens, M.L. and Carvalho, M.M. 2016) are:

- Algorithmic - parametric models.
- Expert Judgment
- Analogy
- Parkinson's
- Price to Win
- Top-Down
- Bottom-Up

Based on this, the author proposes seven basic steps in estimating the cost of software projects:

1. Establish Objectives
2. Plan for Required Data and resources
3. Pin down software requirements
4. Work out as much detail as feasible
5. Use several independent techniques and sources
6. Compare and iterate estimates
7. Follow up

B. Boehm in his work now most cited in this paper "Software Engineering Economics" writes "without data on productivity it is impossible to calculate the cost of a project because it (productivity) is the cornerstone or basis for any calculation (estimation) of the project".

Software Project Constructive Cost Modeling – COCOMO

In order for the investment decision to be as fair as possible, it is necessary to present the most accurate forecast of the costs of the development of the software project, these costs, to the greatest extent, are costs for personnel engagement. So at the beginning of every software project, questions such as:

- How high will be the overall project work costs,
- How long (Person Months) the project will last,
- What should be the number of engaged personnel,
- How much the project will ultimately cost.

In order for the method in question to be as efficient as possible, which means as accurate as possible in the evaluation, a distinction is made between types or models of projects (project modes).

Table 4. Software project cost modeling

COCOMO simple organic model	COCOMO organic model	COCOMO intermediar - semidetached mode	COCOMO detal - embedded mode
Small projects, respectively smaller than 50 000 DSI with a small number of connections (interfaces) with other systems (Stand Alone applications), stable environment for development, which means that every collaborator knows the whole project, respectively has experience in similar projects.	Projects with an average size between 50 000 and 300 000 DSI (delivered source instructions), each collaborator possesses special knowledge related to the development of the project in question, the team that is committed to the development of the project is not trained together, respectively in terms of projects with a degree of complexity or difficulty between Organic and Embedded Mode.	Projects with an average size between 50 000 and 300 000 DSI (delivered source instructions), each collaborator possesses special knowledge related to the development of the project in question, the team that is engaged for the development of the project is not trained together, respectively in terms of projects with a degree of complexity or difficulty between Organic and Embedded Mode	Large projects – over 300 000 DSI, high level of definition in development and the main characteristic is the close connection of the developed software with other hardware-software systems, e.g. flight control software systems and real-time systems etc.

COCOMO's life cycle consists of five stages and that

- Planning and definition of requirements,
- Product design,
- Detailed design,
- Coding and testing of modules,
- Integration and testing.

COCOMO – Base Model

Based on what was said above, the project costs in Person Months can be calculated based on the formula presented below, which the author of this method has arrived at after analyzing 63 completed projects.

PM=2.4(KDSI)1.05

while the duration per month (TDEV-time for development) of the project development is calculated according to the formula

TDEV=2.5(PM)0.38

which derives from the calculated expenses, e.g. if we will develop a software with 50 KDSI according to the above formula we will earn an expense of

PM = 2.4 * (50)1.05 = 145.9 Person Months and with a duration of
TDEV = 2.5 * (145.9) 0.38 = 16.6 Months

Also, with the above-mentioned formulas, we can, in addition to the expenses per Person Month and the duration of the project development, also calculate the productivity of the project development (expressed in lines of the source code for the entire project development process - DSI), if the we related the volume of the project in KDSI with the project development costs.

KDSI / PM = KDSI / 2.4(KDSI)1.05

and also besides the calculated productivity, e.g. for a volume of 50 KDSI with an expenditure of 145.9 PM with a duration of 16.6 months, we can calculate the number of personnel needed since this formula as an average comes out

PM=145.9PM / 16.6 Months = 8.8 People

To calculate the costs and duration of the software project development, CoCoMo uses the same formula for all three types of projects, but with different coefficients and exponents respectively (Table 2) and for this very reason the classification of the project is needed, which is the object of assessment, in one of the types of projects mentioned above.

Table 5. Formulas for calculating expenses and duration by project type

Project type	Expenses	Timeline
Organic	$PM=2.4(KDSI)^{1.05}$	$TDEV = 2.5(PM)^{0.3}$
Partialy integrated	$PM=3.0(KDSI)^{1.12}$	$TDEV =2.5(PM)^{0.35}$

As can be seen in partially integrated and detailed projects, development costs compared to Organic projects will be greater, thanks to the larger exponents and multipliers that are based on them, and as a result, they will have different costs for projects with the same volume and all this based on the type of project in question.

Table 6. Software project characteristics

Project volume = 100KDSI	Project volume = 100KDSI	Project volume = 100KDSI
Type = Organik $PM=2.4(100)^{1.05}$ = 302 PM	Lloji = partialy integrated $PM=3.0(100)^{1.12}$ = 521 PM	Type = Detailed $PM=3.6(100)^{1.20}$ = 904 PM

For the same volume, for partially integrated projects, the duration will be 23.3 and for the Detail 40.9. As we see from the big differences it brings in the evaluation method, the classification in different types of software projects results that, the more precise we will be in respecting the classification criteria, the more accurate our evaluations will be and in after all, this assessment will largely define the fate of the project.

Table 7. COCOMO base model

Phases	Products in KDSI	Participation	Request analysis	Product design	Programing	Testing plan	Verification	Project office	Project quality	Guide formulation
Plans and requests	7	8	48	14	2		2	6	16	5
	8	8	48	13	24	3	7	14	4	7
	32	8	46	14	6	4	8	12	4	6
	128	8	44	15	8	5	9	10	4	5
	512	8	42	16	10	6	10	8	3	5
Product design	2	18	10	42	10	4	6	15	4	9
	8	18	10	42	11	5	13	3	9	7
	32	18	10	42	12	6	8	11	3	8
	128	18	10	42	13	7	9	9	3	7
	512	18	10	42	14	8	10	7	2	7
Programing	2	60	3	6	55	4	8	9	8	7
	8	57	3	6	55	5	9	8	7	7
	32	54	3	6	55	6	10	7	7	6
	128	51	3	6	55	7	11	6	7	5
	512	48	3	6	55	8	12	5	6	5
Integration and testing	2	22	2	4	32	3	30	10	10	9
	8	25	2	4	36	3	28	9	9	9
	32	28	2	4	40	4	25	8	9	8
	128	31	2	4	44	4	23	7	9	7
	512	34	2	4	48	5	20	6	8	7

COCOMO – Intermediar Model

The intermediate CoCoMo model is an extension of the basic CoCoMo model. In the intermediate CoCoMo model, a total of 15 factors are considered (subject to a category that can be very low, low, normal high, very high and extra or extremely high), different in terms of influence, which are summarized in four categories or groups that too

Table 8. COCOMO intermediar model

Features of the software product	**Computer features**	**Personnel characteristics**	**Project characteristics**
RELY- required software reliability - siguria e kërkuar nga softueri, DATA- size of application database – vëllimi i bazës së të dhënave, CPLX – complexity of the product – kompleksiteti i produktit.	TIME – execution time constraint - definitions related to execution time, STOR - main storage constraints - definitions related to the main (operational) memory, VIRT - virtual machine volatility - the stability of the virtual machine, TURN – computer turnaround time – the time needed to restore the computer state	ACAP - analyst capability - system analyst capability, AEXP - applications experience - experience with the application, PCAP - programmer capability - the ability of the programmer, VEXP - virtual machine experience - experience with the virtual machine, LEXP - programming language experience - experience with the programming language.	MDOP - modern programming practices - the application of modern programming practices, TOOL - use of software tools - use of software tools for development, SCED - required development schedule - the appropriate duration of the project.

Table 9. Multipliers for constructive expenditure modeling – intermediate CoCoMo model

Features of the software product	**Computer features**	**Personnel characteristics**	**Project characteristics**
RELY-security required by software 0.75 0.88 1.00 1.15 1.40	TIME–executable time definitions. 1.00 1.11 1.30 1.66	ACAP–system analyzer ability 1.46 1.19 1.00 0.86 0.71	MDOP–application of modern programming language.
DATA–database volume 0.94 1.00 1.08 1.16	STOR–main (operational) memory related definitions	AEXP – experience with the application 1.29 1.13 1.00 0.91 0.82	1.24 1.10 1.00 0.91 0.82 TOOL – use of soft tools. for development
CPLX–product complex 0.70 0.85 1.00 1.15 1.30 1.65	1.00 1.06 1.21 1.56 VIRT – virtual machine stability 0.87 1.00 1.15 1.30	PCAP – programmer skill 1.42 1.17 1.00 0.86 0.70 VEXP – virtual machine experience	1.24 1.10 1.00 0.91 0.83 SCED – appropriate project duration
	TURN – the time needed to restore the state of the computer	1.21 1.10 1.00 0.90 LEXP – programming language experience	1.23 1.08 1.00 1.04 1.10
	0.87 1.00 1.07 1.15	1.14 1.07 1.00 0.95	

Corrected Expenditure Formulas

Project type	basic model	between model
Organic	PM=2.4(KDSI)1.05	PM=3.2(KDSI)1.05
Partially integrated	PM=3.0(KDSI)1.12	PM=3.0(KDSI)1.12
Detail	PM=3.6(KDSI)1.20	PM=2.8(KDSI)1.20

COCOMO –Final Model

The detailed model of the constructive expenditure modeling method differs from the intermediate model of the same method, respectively it is further detailed in two aspects; also in dividing the software project into a three-level hierarchy starting from the lowest, the level of modules, the level of subsystems (subsystems) and the system level.

Table 10. Attributes with their multiplier values by stages at module level

		Clasification					
Atributes	**Phases**	Very low	Low	Norml	High	Very high	Ekstra high
CPLX	Product design	0,70	0,85	1,00	1,15	1,30	1,65
	Detal design	0,70	0,85	1,00	1,15	1,30	1,65
	Coding	0,70	0,85	1,00	1,15	1,30	1,65
	Integration and testing	0,70	0,85	1,00	1,15	1,30	1,65
PCAP	Product design	1,00	1,00	1,00	1,00	1,00	
	Detal design	1,50	1,20	1,00	0,83	0,65	
	Coding	1,50	1,20	1,00	0,83	0,65	
	Integration and testing	1,50	1,20	1,00	0,83	0,65	
VEXP	Product design	1,10	1,05	1,00	0,90		
	Detal design	1,10	1,05	1,00	0,90		
	Coding	1,30	1,15	1,00	0,90		
	Integration and testing	1,30	1,15	1,00	0,90		
LEXP	Product design	1,02	1,00	1,00	1,00		
	Detal design	1,10	1,05	1,00	0,98		
	Coding	1,20	1,10	1,00	0,92		
	Integration and testing	1,20	1,10	1,00	0,92		

After dividing the project into three hierarchical levels, for each of them an assessment is made of their volume in DSI as well as of their attributes such as: CPLX - module complexity, PCAP - programmer skill, VEXP - experience with the virtual machine as well as LEXP – experience with programming language as well as in

case of partial development of the project also AAF and EDSI. Below are presented the attributes with the value of their multipliers according to the phases at the module level, while there is also an analogous table for the attributes located at the subsystem level.

Software Maintenance

The method of constructive expenditure modeling also enables the evaluation of the maintenance of software projects which is defined through the so-called annual change traffic ACT, i.e. we relate the participation of the volume of the source code of the developed software which during a year has undergone changes, respectively has been modified through introducing a new code or changing the existing one. EG a software project with a volume of 32 KDSI (32000DSI) has been developed, where the annual flow of changes is 5,000 new program lines are introduced, while 3,000 program lines are modified from the existing project, then we will have

ACT = 5000+3000*/32000=0.25 and now starting from the project development expenditure PMD (DEVELOPMENT) we calculate the annual maintenance expenditure PMAM (ANNUAL MAINTACE) through the formula

$$\mathbf{PM_{AM} = ACT * PM_D}$$

From the formula in question, we obtain the project maintenance costs per Person Month and for the whole year, and if we want to see the necessary number of personnel for maintenance - full-time software persons, then the amount obtained from the above formula must be divided by 12 month.

FSPM = PMAM/12 months

Table 11. RELY with changed values of multipliers for the maintenance phase

Categorisation	Multiplicators value per development phase	Multiplicators value er maintenance phase
Very low	0,75	1,35
Low	0,88	1,15
Normal	1,00	1,00
High	1,15	0,98
Very high	1,40	1,10

Below in the table are presented multipliers with changed values for MDOP during the maintenance phase.

Table 12. MDOP with changed multiplier values for the maintenance phase

Categorisation	Multiplicators value per development phase	Multiplicators value per maintenance phase				
		2 KIDS	8KIDS	32KIDS	128KIDS	512KIDS
Very low	1.24	1,25	1.30	1.35	1.40	1,45
Low	1.10	1.12	1.14	1.16	1.18	1,20
Normal	1,00	1,00	1,00	1,00	1,00	1,00
High	0.91	0.90	0.88	0.86	0.85	0,84
Very high	0.82	0.81	0.77	0.74	0.72	1,70

As we saw, the CoCoMo method of constructive expenditure modeling enables us, through different levels of detailing, to evaluate the software project, even at the macro level, through the fundamental or basic model, while through the in-between model and that detail we can also do assessment at the micro level. This method in its assessments takes into consideration the characteristics or attributes of the product, personnel, technology as well as the project, while for the objective evaluation of these influential factors the method in question gives us clear instructions even though the quantification of these influential factors does not follow without problems.

Testing the Software

Software and quality testing have the task of finding errors, respectively to see how well the quality requirements have been met. The importance of testing is best seen from the results presented on an empirical basis by Boehm42, according to which the costs of eliminating an error that is discovered after the delivery of the software are on average 100 times greater than when the error is discovered already at the analysis stage.

There are various methods for software testing. The main characteristic of static testing methods is that during testing, the software component is not launched, but the source code is analyzed to find errors, while the so-called manual testing methods such as inspections, reviews and walkthroughs are most often used. Testing is carried out by random selection. The purpose of dynamic testing methods is to detect differences between the actual operation of the developed software and the proper operation according to the specification. in the literature, the Black-Box and White-Box methods are most often mentioned as such testing methods. In Black-Box testing, the test software is treated as a " black box ". The tester is not interested in behavior and internal structure, but his area of interest is to find such circumstances that influence the behavior of the software to deviate from its proper behavior, that which is defined in the specification.Whereas in White-Box testing, the tester's interest is focused on the internal structure of the software being tested, and all of this considering the appropriate state or required by the specification. The goal is therefore to find errors in the program, while the testing is done with those data that will go through all the possible paths (branches) of the program. In conclusion, we can say that ensuring software quality is necessary from many aspects, but when it comes to commercial software, it (quality) ensures competitive (comparative) advantages just as the existence itself, or rather the presence of software in a product compared to the same product but without software e.g. automotive industry.

Research Methodology for Managing Software Projects in Organizations

First we have definen dhe research objectives where clearly articulated the goals and objectives of the research, such as improving software project management practices within trade organizations, enhancing efficiency, or reducing costs. Secondly we did the literature review, conducting a thorough literature review to understand existing methodologies and best practices in software project management, especially in the context of trade organizations. Identify relevant theories, frameworks, and case studies. İn next step the reserach has identified trade organization requirements. Gathering insights into the specific needs, challenges, and constraints faced by trade organizations regarding software project management. This involved interviews, surveys, or focus groups with stakeholders within these organizations.

Based on those we choose the appropriate research methodology based on the objectives, such as quantitative, qualitative, and mixed-method approaches. We considered case studies, surveys, interviews, and did design data collection instruments tailored to the chosen methodology. Before finalizing gthe methodology we did conduct a pilot test of the research instruments to ensure they are effective and valid. İn next step we did implement the research plan to collect relevant data from organizations. As well we ensure data collection methods align with ethical standards and address any privacy or confidentiality concerns. Further th reserach did analyze the collected data using appropriate techniques based on the research questions and methodology. This involved quantitative analysis (e.g., statistical tests, regression analysis) and/or qualitative analysis (e.g., thematic analysis, content analysis) followed by software development acording to requests of organization and finalized by software testing. The findings of the research was very promising and the developed and adopted sofware for the choosen company has function well. İn the end of research the recommendations and conclusions based on the findings, will be drawn regarding the management of software projects in organizations. The paper will provide practical recommendations for improving software project management practices, addressing challenges, and leveraging opportunities.

Case Study for This Research

In recent decades, the software industry and its products have become one of the factors without which the normal functioning of contemporary businesses and economies cannot be imagined. Therefore, it is difficult to imagine any company that claims to be competitive, without software support, whether it is standard software or developed specifically for the needs of the company. The paper has as a case study the development and evaluation of a software project in the company Elkos Group and describes the steps of how a software project is developed and implemented for the needs of the company.

Development of the Human Resources Software Project at Elkos Group

Considering the large capacity of work and the large number of workers that the Elkos Group company has, the need has arisen for the design and development of this project in order to increase work efficiency and for the administration in this company to be as functional as possible. With the implementation of this project, it will be

possible to manage the company's staff. With the implementation of this application, administrative procedures will be significantly reduced, thereby increasing work efficiency, we will have a clear overview of the company's staff.

Technical Solution and Description of the Database

This system enables easy and efficient management as well as easy services of Human Resources in the Company. The database of this application consists of several tables and their diagram looks like the following.

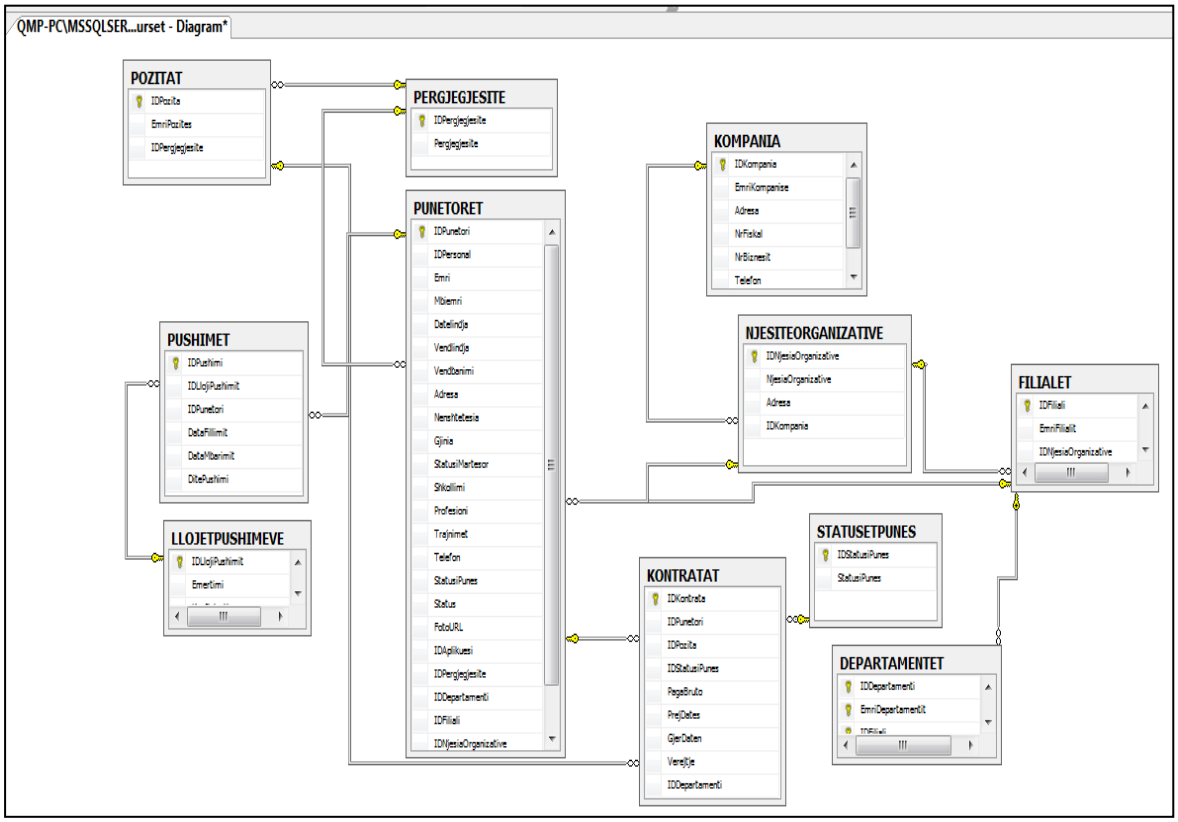

Figure 6. Human resources application software description

Table 13. Digitization and Registration of Departments

Field	Type of data	Description
ID Departament	int	Unique number that identifies the registered apartment and which is automatically increased by the system
Departament name	varchar (50)	Describes the Name of the Apartment
ID Branch	int	Unique number identifying the registered Fiialit

Table 14. Digitization and registration of subsidiaries

Field	Type of data	Description
ID Branch	int	Unique number identifying the registered Affiliate
Branch name	varchar (50)	Describes the Affiliate Name
ID org, unit	int	Unique number that identifies the registered Organizational Entity

Table 15. Digitization of the company

Field	Type of data	description
ID Company	int	Unique number that identifies the registered Company
Company name	varchar (50)	Describes the Company Name
Adress	varchar (50)	Describes the address of the Company
Fiscal nr	varchar (50)	Describes the fiscal number of the Company
Bussines nr	varchar (50)	Description of the Company's business number
Telefon	varchar(50)	Describes the Company's phone number

Table 16. Digitizing the registration of employee contracts

Field	Type of data	Description
ID Contrats	int	Unique number that identifies the registered contract and which is automatically increased by the system
ID Employee	int	Unique number that identifies the registered employee and which is automatically increased by the system
ID Position	int	Unique number that identifies the position of the registered employee and which is automatically increased by the system
ID Status	int	Unique number that identifies the status of the registered employee
Wage Bruto	float	It shows the gross payment that came out of the contract
From	datetime	Describes the start date of the contract
To	datetime	Describes the End Date of the contract
Remarks	text	Describe the murder, why the contract is over
ID Departament	int	Unique number that identifies the registered apartment and which is automatically increased by the system

Table 17. Digitizing the registration of the type of employees' vacations

Field	Type of data	Description
ID vacation type	int	Unique number that identifies the Type of Holiday registered and which is automatically increased by the system
Designation	varchar (50)	Describes the Vacation Type
Coefficient	float	Describes the coefficient of rest

Table 18. Digitization of registration of organizational units

Field	Type of data	Description
ID org. Unit	int	Unique number that identifies the registered Organizational Unit
Org. Unit	varchar (50)	Describes the Designation of the Organizational Unit
Adress	varchar (50)	Describes the address of the organizational unit
ID Company	int	Describes the ID of the Company in which the Organizational Unit is registered

Table 19. Digitizing the registration of human resources employees

Field	Type of data	Description
Worker ID	int	Unique number that identifies the registered employee and which is automatically increased by the system
Personal ID	varchar(50)	Describes the employee's personal number
Name	varchar(50)	Shows the name of the employee
LAST	varchar(50)	Shows the last name of the employee
Birthday	datetime	Shows the date
Birthplace	varchar(50)	It shows the place of birth
RESIDENCE	varchar(50)	Shows Residence
address	varchar(50)	Shows the address
Citizenship	varchar(50)	Shows the citizen
gender	varchar(50)	Describes gender
Marital Status	varchar(50)	Describes the marital status of the employee
SCHOOLING	varchar(50)	Describe the worker's education
OCCUPATION	varchar(50)	Describes the profession of worker
Training	varchar(50)	Describes the training that the employer has completed
phone	varchar(50)	Shows the phone number of the employee
Job Status	varchar(50)	Describes whether it is part-time or full-time
Status	varchar(50)	It describes the status whether it is Active or Passive
Photo URL	varchar(50)	It shows where the employee's photo is stored
Applicant ID	int	The unique number of the applicant who applied for the job
Responsibilities ID	int	The unique number of the employee's responsibilities
Department ID	int	Unique number that identifies the apartment where the employee works
Affiliate ID	int	Unique number that identifies the Branch where the employee works
Organizational Unit ID	int	Unique number that identifies the Organizational Unit where the employee works
Status	varchar(50)	It describes whether it is a worker or an applicant

Discusion and Description

This section demonstrates the key findings shown from the conducted investigations and will discuss the generated themes. In particular, to answer the research question about what organizations can do to enable software project systems drive their businesses towards growth and sustainability, we will show the activities undertaken by the firm taken as case study that assumed to be effective for software system role that enables business growth and sustainability. These activities are drawn based on the sustainability principles that have been suggested by Huemann and Silvius (2017). The Human Resources software management project ensures the management of Human Resources in a Company, the basic objectives of this Project are:

- Registers and manages the Company with Organizational Units and Branches of Organizational Units
- Registers and manages the Working Staff
- Records and manages Workers' Holidays
- Registers and manages Employee Contracts

Conclusion

Based on the literature review provided, and case study developed here are some conclusions and recommendations regarding software project management and its economic evaluation in terms of enterprise sustainability:

Integration of Sustainability into Project Management: There is a growing recognition of the importance of integrating sustainability considerations into software project management practices. This integration involves aligning project management frameworks and methodologies with broader sustainability goals to ensure the long-term viability and resilience of the organization.

Economic Evaluation for Sustainability: Traditional economic evaluation techniques, such as Cost-Benefit Analysis (CBA) and Return on Investment (ROI), need to be augmented to account for sustainability factors. Organizations should consider the economic impact of sustainable software development practices, such as resource efficiency and waste reduction, in their project evaluation processes.

Importance of Sustainable Practices: Adopting sustainable software development practices not only contributes to environmental sustainability but also offers economic benefits. Practices such as green computing and energy-efficient programming can lead to cost savings through reduced resource consumption and improved operational efficiency.

Challenges in Implementation: Despite the potential benefits, organizations face challenges in implementing sustainable software project management practices. These challenges include resistance to change, lack of awareness or understanding of sustainability issues, and the need for investment in new technologies and skill development.

Recommendations

Develop Comprehensive Sustainability Frameworks: Organizations should develop comprehensive frameworks and methodologies for integrating sustainability into software project management practices. These frameworks should provide guidance on incorporating sustainability considerations into project planning, execution, and evaluation processes.

Educate and Train Project Managers: Project managers play a crucial role in driving the adoption of sustainable software development practices. Organizations should invest in training and education programs to equip project managers with the knowledge and skills needed to integrate sustainability into their project management approach.

Incentivize Sustainable Practices: Organizations can incentivize the adoption of sustainable software development practices by incorporating sustainability metrics into project evaluation criteria. Recognizing and rewarding teams that achieve sustainability goals can help foster a culture of sustainability within the organization.

Collaborate with Stakeholders: Collaboration with stakeholders, including customers, suppliers, and regulatory bodies, is essential for advancing sustainability in software project management. Organizations should engage with stakeholders to identify sustainability priorities, share best practices, and address common challenges.

Invest in Research and Development: Continued research and development are needed to advance sustainable software development practices and technologies. Organizations should invest in R&D efforts aimed at developing innovative solutions for minimizing the environmental impact of software projects while maximizing economic value.

In conclusion, integrating sustainability into software project management practices requires a multi-faceted approach that considers economic, environmental, and social factors. By adopting sustainable practices, organizations can not only contribute to a more sustainable future but also realize economic benefits and enhance their long-term competitiveness

Scientific Ethics Declaration

The authors declare that the scientific ethical and legal responsibility of this article published in EPSTEM journal belongs to the authors.

Acknowledgements or Notes

* This article was presented as an oral presentation at the International Conference on Basic Sciences, Engineering and Technology (www.icbaset.net) held in Alanya/Turkey on May 02-05, 2024.

References

Aarseth, W., Ahola, T., Aaltonen, K., Økland, A., & Andersen, B. (2017). Project sustainability strategies: A systematic literature review. *International Journal of Project Management*, *35*(6), 1071-1083.

Acar, H. (2017). *Software development methodology in a Green IT environment* (Doctoral dissertation, Université de Lyon).

Albertao, F., Xiao, J., Tian, C., Lu, Y., Zhang, K. Q., & Liu, C. (2010). Measuring the sustainability performance of software projects. In *2010 IEEE 7th International Conference on E-Business Engineering* (pp. 369-373). IEEE.

Anaya, L. (2019). To what extent is it viable to apply benefits management approach for ERP systems?. *Procedia Computer Science*, *164*, 33-38.

Alharthi, A., Spichkova, M. & Hamilton, M. (2016), "Sustainability profiling of long-living software systems", in QuASoQ 2016: International Workshop on Quantitative Approaches to Software Quality, CEUR-WS, (pp. 12-19).

Becker, C., Betz, S., Chitchyan, R., Duboc, L., Easterbrook, S.M., Penzenstadler, B., Seyff, N. & Venters, C.C. (2016). Requirements: the key to sustainability. *IEEE Software, 33*(1), 56-65.

Becker, C., Betz, S., Chitchyan, R., Duboc, L., Easterbrook, S. M., Penzenstadler, B., & Venters, C. C. (2015). Requirements: The key to sustainability. *IEEE Software*, *33*(1), 56-65.

Beghoura, M.A., Boubetra, A. and Boukerram, A. (2017). Green software requirements and measurement: random decision forests-based software energy consumption profiling, *Requirements Engineering, 22*(1), 27-40.

Calero, C., & Piattini, M. (2017). Puzzling out software sustainability. *Sustainable Computing: Informatics and Systems*, *16*, 117-124.

Calero, C., Moraga, M., & Bertoa, M. F. (2013). Towards a software product sustainability model. *arXiv preprint arXiv:1309.1640*.

García-Mireles, G. A., Moraga, M. Á., García, F., Calero, C., & Piattini, M. (2018). Interactions between environmental sustainability goals and software product quality: A mapping study. *Information and Software Technology*, *95*, 108-129.

Huemann, M., & Silvius, G. (2017). Projects to create the future: Managing projects meets sustainable development. *International Journal of Project Management*, *35*(6), 1066-1070.

Kern, E., Naumann, S., & Dick, M. (2015). Processes for green and sustainable software engineering. *Green in Software Engineering*, 61-81.

Lago, P., Koçak, S. A., Crnkovic, I., & Penzenstadler, B. (2015). Framing sustainability as a property of software quality. *Communications of the ACM*, *58*(10), 70-78.

Marnewick, C. (2017). Information system project's sustainability capabality levels. *International Journal of Project Management*, *35*(6), 1151-1166.

Martens, M.L. and Carvalho, M.M. (2016a), "Sustainability and success variables in the project management context: an expert panel", Pro Martens, M. L., & Carvalho, M. M. (2016). Sustainability and success variables in the project management context: an expert panel. *Project Management Journal*, *47*(6), 24-43.

Økland, A. (2015). Gap analysis for incorporating sustainability in project management. *Procedia Computer Science*, *64*, 103-109.

Pohludka, M., Stverkova, H., & Ślusarczyk, B. (2018). Implementation and unification of the ERP system in a global company as a strategic decision for sustainable entrepreneurship. *Sustainability*, *10*(8), 2916.

Savitz, A. (2013). *The triple bottom line: how today's best-run companies are achieving economic, social and environmental success-and how you can too*. John Wiley & Sons.

Schieg, M. (2009). The model of corporate social responsibility in project management. *Verslas: Teorija ir Praktika*, *(4)*, 315-321.

Silvius, A. G., Kampinga, M., Paniagua, S., & Mooi, H. (2017). Considering sustainability in project management decision making: An investigation using Q-methodology. *International Journal of Project Management*, *35*(6), 1133-1150.

Zerbino, P., Aloini, D., Dulmin, R., & Mininno, V. (2021). Why enterprise resource planning initiatives do succeed in the long run: A case-based causal network. *Plos one*, *16*(12), e0260798.

Turner, J.R., Anbari, F. and Bredillet, C. (2013). Perspectives on research in project management: The nine schools, *Global Business Perspectives*, *1*(1), 3-28.

Yunis, M., Tarhini, A., & Kassar, A. (2018). The role of ICT and innovation in enhancing organizational performance: The catalysing effect of corporate entrepreneurship. *Journal of Business Research*, *88*, 344-356.

Zerbino, P., Aloini, D., Dulmin, R., & Mininno, V. (2021). Why enterprise resource planning initiatives do succeed in the long run: A case-based causal network. *Plos one*, *16*(12), e0260798.

Ziemba, E. (2018). The contribution of ICT adoption to sustainability: Households' perspective. *Information Technology & People*, *32*(3), 731-753.

Author Information

Ibrahim Krasniqi
University Haxhi Zeka
Rr. UCK, nn, 30000 Peje, Kosovo
Contact e-mail: *ibrahim.krasniqi@unhz.eu*

Naim Ismajli
AAB College Prishtina, Kosovo

Ganimete Krasniqi
MMPHI-GoK, Rilindja Building, Prishtina, Kosovo

To cite this article:

Krasniqi, I., Ismajli, N., & Krasniqi, G. (2024). Software project management and their economic evaluation in terms of enterprise sustainability, *The Eurasia Proceedings of Science, Technology, Engineering & Mathematics (EPSTEM), 28*, 533-553.