Analysis of Software Project complexity factors of large scale systems and their impacts on core knowledge areas of software project Management

Mehreen Sirshar
Faculty of Software Engineering
Fatima Jinnah Women University
Rawalpindi, Pakistan
mehreensirshar@fjwu.edu.pk

Sara Ibrahim
Department of Software Engineering
Fatima Jinnah Women University
Rawalpindi, Pakistan
saraibrahim7479@gmail.com

Asma Ali
Department of Software Engineering
Fatima Jinnah Women University
Rawalpindi, Pakistan
aasma6038@gmail.com

***Abstract*— Software project complexity increases day by day because the software engineering products is being used in the solution of more technically difficult problem and the size of project continuous to grow. The increase complexity causes to high numbers of software project failures in term of time, cost and quality. The main question regarding to this problem is how to handle or cope with this complexity. There is no single way to handle this, software engineer uses different perspective to handle complexity without affecting the overall project performance. A management perspective recognizes that the success of complex project requires good project management. A technically perspective reveals new paradigms for software development i.e.; object oriented and formal methods etc. and software engineer also look for automation perspective in order to reduce the complexity issues. In this paper we will find out the main software project complexity factors by focusing on the management aspects of software project development and also the problems of managing complexity in software engineering products from these different perspectives. The paper is divided in three main sections; paradigms of software development, project management in term of time, cost and quality and third one is automated support that includes methods and tools used to manage the complexity.**

I. INTRODUCTION

Software complexity is difficult to identify because it's related to overall developing process. Complexity of software defined as the degree of difficulty in computer evaluation, maintenance, testing, development and alteration. Complexity is, according to the Lachmannian process theory, the sum of the following components: distinction of functions among project participants, system and subsystems dependency, and the consequent impact of a decision area. Custovic describes complexity as the property of a system that makes it difficult to articulate its overall behavior in a particular language, even if it provides appropriate complete information about its atomic components and their interrelations.

The importance of project complexity to the project management system is widely recognized for several reasons: I it affects project scheduling, coordination, and control; (ii) it impedes clear identification of major project goals and objectives; (iii) it may affect the choice of suitable project organization type and management staff experience requirements; (iv) may be used as criteria for selecting an appropriate project management arrangement; and (v) it may affect the various project outcomes (time, cost, quality, safety, etc.). As projects become more and more complex, there has been a need to develop such tools and techniques that will helpful in managing the project complexity. According to Parsons-Hann and Liu, it is obvious that uncertainty leads to organizational task failures; what is not clear in the given requirements is directly proportional to what degree these statements are true. So Identifying the basic factors of complexity and characterizing different aspects of project complexity in order to better understand the risk factor of the complexity of project management and it can be a great support in assisting the community of project management. Project complexity has two types of influence on the project positive and negative effect. Due to the emergence of new properties owned by none of the elements of the system, the negative influence leads to difficulty in understanding and controlling the project. And the positive aspect of project complexity is that more complex and large scale projects have great chance of success because it covers the more organizational needs.

Complexity is a broader term that can't be easily defined, two main categories of complexity is "**Complexity of the problem";** that is basically the inherent complexity, and introduced during the phase of requirement gathering. Problem complexity also determines that amount of resources needed to solve the problem optimally. "**The solution Complexity";** also referred to as added complexity attached to the problem's complexity. This type of complexity was added to the design and coding phases during the development phase following the requirement phase.

Structural and Algorithm complexity are also types of complexity the characteristic measure of structural is error proneness and for algorithm complexity it is size and effort. The tract of error proneness leads to software quality and size/effort to software productivity.

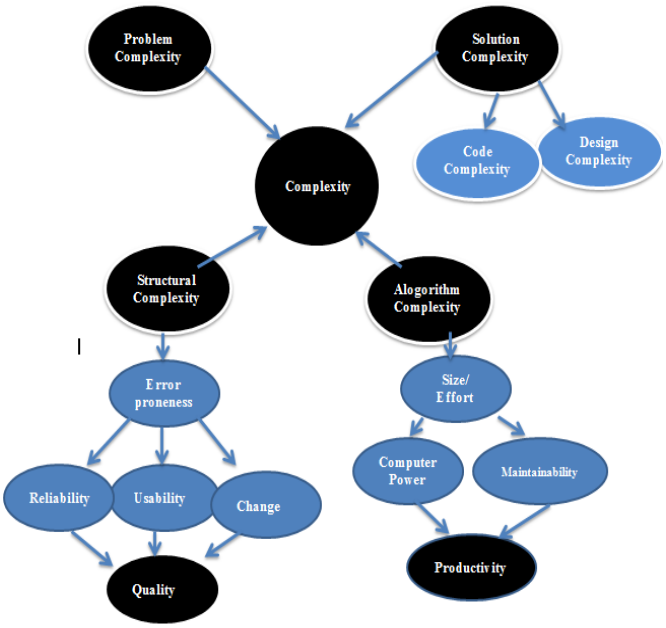

Figure 1: A model of software Complexity

In this paper we will discuss the main causes or types of complexity, how they occurred and in what phase of developing, how to manage the complexity and what modern tools and techniques used to handle the more complex projects and at the end we also discuss how to increase successful rate of complex projects.

II. Literature Review

In [1] Rafal Kozik et al., describe that complexity of software engineering products and software quality assurance has dramatically increased. They present methodology and tools to collect and analyze information related to the development of software projects. To improve the rapid process of software development, they use "machine learning, data analysis and visualization technique". They use metrics, bug tracking and testing to measure the software quality in term of cost and time.

F.L. Gaol et al., [2] uses artificial neural network for predication of software complexity based on complex requirement. One of the main factors of complexity is complex requirement as they are origin so in the result complex system produce.so they use artificial neural and Bayesian networks to predict the complexity of requirements and also make comparison of these two algorithms.

Balikuddembe et al., [3] provide the analysis that computer implementation in the public sector use the value based technology techniques. They examine in their research what are the key issues and gap that need to be addressed in order to deal with requirement complexity issues for large scale projects. They also highlighting that the information about business perspective, products perspective and then project perspective are very important because these perspective provides the better understanding of projects.

Davoud Mougouei [4] uses graph-based dependency modeling techniques to capture dependencies from specifications and their corresponding strengths while holding costs according to the budget. They also describe that existing selection models ignored the requirement dependencies and their strengths values that causes the complexity issues in requirement and then leads to complexity within software projects that causes projects failures.

Rosa et al.., [5] describe that software effort are necessary and very critical for decision making at initial stages for initial budget estimation, schedule and for defining the requirements. They use agile approach for software development and use statistical analysis for measurement of complexity factors in term of cost, time and quality.

In [6] Prause made their research in four domain from aerospace to research, in order to extract some of the hidden expert knowledge express through the usage of process. It analyze data from a survey in a biotope of un regulated projects and three is a standard with respects to how to process cost and benefits are balanced through critically tailoring of process. The overall observation shows that higher critical software means high rate of complexity.

In [7] R.V.Sivabalan et al., use the object oriented based architecture that delivers the quality conscious technology infrastructure, including low development costs, proved reusability of code and decreased complexity of software projects.

W.Aslam et al., [8] uses agile software development approach for assigning the appropriate task to particular project member. This approach facilitates the "project management activities, lessens the complexities, and also provide the great the chance of project success. They also use a task allocation framework that consist of two main phases; the first one is used to identifying the factors and dependencies that is strongly effect the task allocation decision; the second one is used to, proposing a quantitative method that allocate the task to each members of team who best match the task requirements. In this way understanding of project increases that reduce the complexities issues within the project development process.

In [9] Johanssen et al., describes the concept of virtualization in continues software engineering. Due to advancement in technology, complexity also increases so processing and accessing the knowledge is challenging for developers. So they purpose the dashboard systems that visualize knowledge from different sources. This will helps the developers to "follow, reflect, interact and react on knowledge" in continues software engineering environment. Springboards also introduce for knowledge selector that give the widgets in visualize form in which developers can spot, range and compare the results in order to improve the understanding and reducing the complexity issues.

A.sharon et al, [10] use the model-based approach to work-break structures for the planning of complex system projects. They claim that the complexity of project depends on the nature of services, number and variety of elements, and the interconnection between them, as well as ambiguity due to the development of modern or super high-tech systems. The complexity of project is usually handled by decomposing its architecture into a hierarchical structure. They use "WBS, Gantt chart, program evaluation and review techniques" in order to handle the complexity of software projects.

Hongjum He et al..,[11] enhance IEPUG's complexity matrix in embedded software projects. IEPUG is widely used as a method of measuring the functional size, but when this method is used in embedded software projects then there is big deviation of measurement results and actual results. They use different method for improvement to adjusting the value range of low complexity functional types, and experimental results shows that the accuracy of the measurement result on the basis of the adjusted complexity matrix is significantly improved.

N.Berthier et al.., [12] the ever-increasing complexity of software systems has contributed to an automated management solution being developed. The careless combination of mangers may leads to inconsistences into taking decision, and other management issues leads to complexity within the project. They also use a new approach based on synchronous programming and discrete controller synthesis techniques for the design of AMSs (automated management systems). They provides us with a high level language for modeling the process to be handled, as well as mechanism for dynamically ensuring that there are no logical synchronization issues.

Q.lei et al.., [13] Complexity of software engineering products continues to increase, so there is need to resolve the performance challenges to deploy thousands of coexisting application to work cooperatively and to reach the requirements of efficiency and scalability of micro services architecture. They use kubemark method for performance testing of IT projects to enhance efficiency and cope with complexity issues in larger IT projects.

T.Dyba et al.., [14] use the agile software approach for controlling and managing the projects. There main focus on four areas of managing projects that are; minimum critical specification, autonomous team, self-management, redundancy, feedback and learning in order to reduce the complexity issues and maximize the rate of successful IT projects.

Sommerville [15] et al, discuss the complexity issues in large scale IT projects. They describe the key issue of software complexity and failure of project is the use of existing systems and assembled into new systems. Inherent and epistemic complexity causes the project failure because of lack of background information of existing system and dynamic relationship between elements in a system.

Rocio et al.., [16] describe that complex projects require specific development of project management skills. Complexity must be assessed quantitatively in line with current PM standards in order to lead complex project to success. Their research's main goal is to target the project management complexity assessment tools that calculate the complexity of IT projects and verify all the complexity factors.

Whyte et al.., [17] describe how to manage the changes in complex project deliverables, as well as configuration management, information on assets and big data. The paper contributes to flexibility in complex projects wher e integrity is important by uncovering limitations. Change management issues are addressed, taking into account the changing complexity of configuration management; future use of analytics on complex objects; and research and practice implications.

B. Ronen et al.., [18] describe the role of project management principles in production management. They also describe that one third of projects are successful, and one third of projects have time, and budget and filed overruns of the desired functionality. They use divide and conquer approach in order with complexity, time and cost issues.

Cristobal et al.., [19] provide an overview of the complexity of the project management team and suggest a number of ideas on how to deal with these issues. Complexity effects the management of project planning and control. They purpose several ideas and tools and techniques for measuring the complexity factors.

Fitsilis et al.., [20] shows that many project fails to meet the requirements in term of time, cost and quality restriction. Due to their special characteristics, increased complexity of modern software projects is the main reason for these failures. They purpose structured project management techniques to handle the complexity. As an effective underling project management structure, they use PMBOX (project management body of knowledge) to define uncertainty factors that impact software projects and also use the metrics to assess the complexity of project scope management.

III. Analysis of Software Project Comlexity Factors

Complexity of software is the degree of difficulty in "software analysis, maintenance, testing, design and modification". Complexities in software project management refer to various challenges in running a software project. The main objective of project management software is to allow a team of developers to work effectively in a given time to complete a project successfully. But project management is a difficult task because complexities of software projects continues to increase, and there is need to understand the factors that introduce complexities in software projects and what are the mechanisms to handle these complexities in order to increase the rate of successful projects.

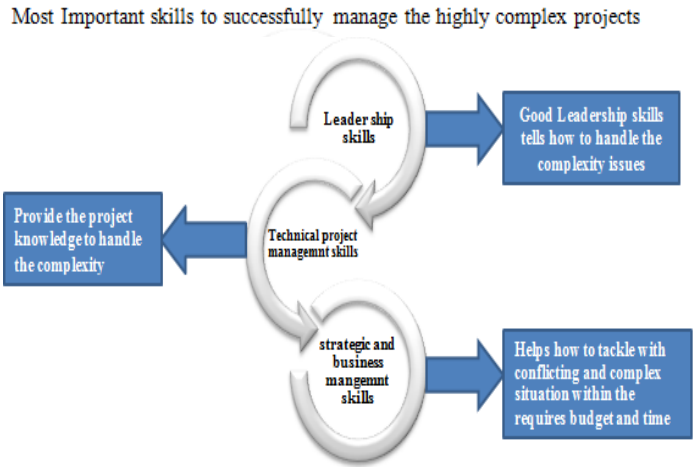

Figure 2: Skills to manage the complexity

There are different types of complexities that introduce in software projects during developing phase.

Time Management complexity: complexities that introduce during the duration estimation of projects. It includes making the schedule of each activity, when it starts and how much time it takes to complete for the timely completion of projects.

Cost Management complexity: It is very difficult to calculate the total cost of projects because you have to calculate the cost of each individual activity and also keep an eye that project does not overrun the budget.

Quality Management Complexity: Quality management is another complex activity to ensure that all the project requirements are according to customer expectation and also within the time, cost and budget.

Risk Management Complexity: Risks are unexpected things that can happen during any phase of the project. These risks can be identified and amended in order to reduce the effects of these risks.

Human Resources Management Complexity: Human resource management is another factor of complexity if it not managed in a proper way. It includes all the difficulties regarding organizing, managing, leading the projects and the gap between top and low order etc.

Communication management complexity: All the members have to communicate with all other members and remove the communication gap between top and lower order because this gap creates inconsistencies within the project.

Procurement Management complexity: Projects needs many services from third party in order to complete the task and this may increase the complexity within the projects to acquire the services.

Integration Management Complexity: compiling the different parts of projects may introduce inconsistency and increase the complexity during software project development.

These are 9 knowledge areas of SPM in which complexity can introduce and now we discuss how to cope with these complexities issues. From a variety of viewpoints, Software engineers address this issue, involving technological, managerial and automation issues.

The technical perspective concern with use of new paradigms for system development i.e.; objects oriented and formal methods etc. The management perspective describes that how good the technical work, success will not have achieved on a complex project unless it is well managed and automation perspective concern the use of automated tools in order to reduce the complexity issues.

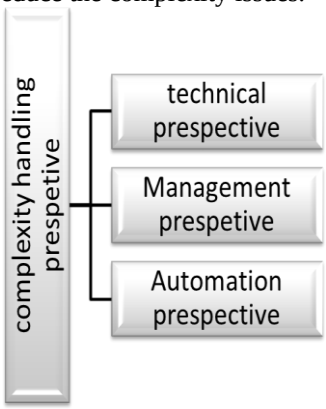

Figure 3: Complexity Handling Prespective

Technical perspective

Encapsulation: encapsulation is a way to hide the information behind the design of the application. It is the important property of object oriented world, encapsulation unifies data model of code and procedural model into obstruction of objects.

Repository architecture: To manage the complexity repository architecture is the most effective way. The repository is database in most of the cases but blackboards are also available. Basically Repository architecture is used for design pattern and for recorded benefits and implications it is highly applicable to system level architecture. It handles complexity by consolidating access through query language or accessor methods to many objects. A query language can distribute messages to thousands of entities and provide a common language template and access protocols to handle a large number of objects.

Horizontal Architecture: Non-essential benefits of complex projects can be overlooked by specifying the standards interface abstraction which provides many advantages such as "interoperability, adaptability, substitutability, and isolation".

Layered Architecture: layered architecture helpful in managing complexity because it determines the abstraction level of the system, in order to separate the application software from low-level information. It also sets horizontal services with common interface abstraction so that these services are reused by multiple application objects and high-level service objects. In addition to managing complexity, layering architecture offers many benefits such as software reuse, interoperability and portability benefits.

Vertical Architecture: Within separate subdomains, vertical partitions can isolate complexity and use a single vertical framework for each subdomain. Vertical dependencies in the vertical partition can be restricted to objects. Since vertical partitioning is ineffective without horizontal interfaces, good architecture should have balanced between horizontal and vertical interfaces.

Management perspective

The goal of software project manager is to produce the good quality project that is within time, cost and budget. But in order to achieve the high quality products and to deal with complexity issues quality management culture must be developed within the company. Both the technical and non-technical skills must be properly measured within the developing team members. The communication gap between top order and low level people must be minimizing. Interaction and communicational skills between stakeholders and development teams must be clear in order to produce the good technical work because success will not have achieved on a complex project until it is well managed.

Automation perspective

Many automated tools used for project evolution to measure the complexity of projects.

PMBOX: project management body of knowledge has been used as standard by which PMP certification is obtained. PMBOX allow the companies to standardize the practices across the department. The method documented within the project management community can assist how to handle the risk and complexity within the projects. PMBOX also tells what does not work so it's helpful to prevent the projects from failures.

Program Evaluation Review Technique (PERT): PERT is a tool for scheduling and managing the activities that are needed to complete the project. PERT and critical path process CPM are often used interchangeably, the only distinction between them is the measurement of the task time. This chart displays the whole project schedule in sequence and also tells which activities performed in parallel. A graphical representation of "network diagram" and "CPM diagram" used to reflect graphically the interrelationships of the project components and to display the order in which each operation was carried out.

Gantt chart: to manage the time complexity issues most of the companies used Gantt chart. Gantt chart shows the assignments of calendar time tasks in days, weeks and months and also used graphical representation to shows start, lapse, and completion time of each task of project. Also include the comparison of the number of days required to complete a project that hits a milestone can be compared to the amount expected and projected. This information helps the slippage or failure points of the target timeline.

Complexity Factors of IT Projects

Group of complexity factors	Factors
Objective, Requirements and expectations	Clear statement of requirements Realistic expectations Clear strategic objectives Uncertain and changing Regulatory requirements
Stakeholders, Interested parties, Integration	User Involvement Executive management support Project sponsor committed with project methodology
Leadership, teamwork, decisions	Near shore/ off shore teams involved All the development team and other people involved are familiar with technical and business aspects of projects.
PM Methods, tools and techniques	Incremental or iterative methodology used in the projects
Technology	Incompetence on using / applying technology New technology IT project management support

Table 1: IT project Complexity Factors

IV. Conclusion And Future Work

When problems are fundamentally dynamic are treated statically, then there are growing delays and budget overruns occurs. Traditional methods and strategies of project management have been found to be inefficient, based on the assumptions that task selection can be discrete; well specifying "time, cost and resources data, and detailed pre-planning and command. Such traditional approaches using a static approach include optimistic forecasts to project managers, ignoring various feedback processes and the project's non-liner relationship. Then the interrelationships between the project components are more complex than traditional techniques suggest, making them inadequate for today's dynamic project environment challenges. The new complex and more demanding and continuously changing worlds needs project managers to reconsideration the traditional definition of project concepts and the way to manage them. Project managers need to be able to take decision and manages these continues evolving projects and requirements that are difficult to predict in term of complexity. To achieve this goal, it is necessary to investigate further streamlined project management approaches in complex environments and new project planning, scheduling, execution, and monitoring methods.

V. References

[1] R. Kozik, M. Choraś, D. Puchalski and R. Renk, "Data analysis tool supporting software development process," *2017 IEEE 14th International Scientific Conference on Informatics*, Poprad, 2017, pp. 179-184.
doi: 10.1109/INFORMATICS.2017.8327243

[2] W. M. Purawinata, F. L. Gaol, A. Nugroho and B. S. Abbas, "The prediction of software complexity based on complexity requirement using artificial neural network," *2017 IEEE International Conference on Cybernetics and Computational Intelligence (CyberneticsCom)*, Phuket, 2017, pp. 73-78.
doi: 10.1109/CYBERNETICSCOM.2017.8311687

[3] J. K. Balikuddembe and J. Nakirijja, "Planning for Public Sector Software Projects Using Value-Based Requirements Engineering Techniques: A Research Agenda," *2018 IEEE/ACM Symposium on Software Engineering in Africa (SEiA)*, Gothenburg, 2018, pp. 50-54.

[4] D. Mougouei, "Factoring requirement dependencies in software requirement selection using graphs and integer programming," *2016 31st IEEE/ACM International Conference on Automated Software Engineering (ASE)*, Singapore, 2016, pp. 884-887.

[5] W. Rosa, R. Madachy, B. Clark and B. Boehm, "Early Phase Cost Models for Agile Software Processes in the US DoD," *2017 ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*, Toronto, ON, 2017, pp. 30-37.
doi: 10.1109/ESEM.2017.10

[6] C. R. Prause, "Critical Software Cultures - Analyses of Processes in Four Domains," *2016 IEEE/ACM International Conference on Software and System Processes (ICSSP)*, Austin, TX, 2016, pp. 11-15.
doi: 10.1109/ICSSP.2016.010

[7] R. S. A. Sreekumar and R. V. Sivabalan, "A survival study of object oriented principles on software project development," *2015 Global Conference on Communication Technologies (GCCT)*, Thuckalay, 2015,pp.307-310.
doi: 10.1109/GCCT.2015.7342673

[8] W. Aslam and F. Ijaz, "A Quantitative Framework for Task Allocation in Distributed Agile Software Development," in *IEEE* doi: 10.1109/ACCESS.2018.2803685

[9] J. O. Johanssen, A. Kleebaum, B. Bruegge and B. Paech, "Towards the Visualization of Usage and Decision Knowledge in Continuous

Software Engineering," *2017 IEEE Working Conference on Software Visualization (VISSOFT)*, Shanghai, 2017, pp. 104-108.

[10] A. Sharon and D. Dori, "A Project–Product Model–Based Approach to Planning Work Breakdown Structures of Complex System Projects," in *IEEE Systems Journal*, vol. 9, no. 2, pp. 366-376, June 2015.
doi: 10.1109/JSYST.2013.2297491

[11] Hongjun He, Lei Xia, Li Luo, Huzhong Yan, Jiao Zhu and Jibao Tang, "Improvement of complexity matrix of IFPUG in embedded-software projects," *2016 2nd IEEE International Conference on Computer and Communications (ICCC)*, Chengdu, 2016, pp. 911-914.
doi: 10.1109/CompComm.2016.7924836

[12] N. Berthier, É. Rutten, N. De Palma and S. M. Gueye, "Designing Autonomic Management Systems by Using Reactive Control Techniques," in *IEEE Transactions on Software Engineering*, vol. 42, no. 7, pp. 640-657, 1 July 2016.
doi: 10.1109/TSE.2015.2510004

[13] Q. Lei, W. Liao, Y. Jiang, M. Yang and H. Li, "Performance and Scalability Testing Strategy Based on Kubemark," *2019 IEEE 4th International Conference on Cloud Computing and Big Data Analysis (ICCCBDA)*, Chengdu, China, 2019, pp. 511-516.
doi: 10.1109/ICCCBDA.2019.8725658

[14] T. Dybå and T. Dingsøyr, "Agile Project Management: From Self-Managing Teams to Large-Scale Development," *2015 IEEE/ACM 37th IEEE International Conference on Software Engineering*, Florence, 2015, pp. 945-946.
doi: 10.1109/ICSE.2015.299

[15] Sommerville, Ian & Cliff, Dave & Calinescu, Radu & Keen, Justin & Kelly, Tim & Kwiatkowska, Marta & Mcdermid, John & Paige, Richard. (2018). Large-scale Complex IT Systems. Communications of the ACM. 55. 10.1145/2209249.2209268.

[16] Rocio Poveda-Bautista, Jose-Antonio Diego-Mas, and Diego Leon-Medina, “Measuring the Project Management Complexity: The Case of Information Technology Projects,” Complexity, vol. 2018, Article ID 6058480, 19 pages, 2018. https://doi.org/10.1155/2018/6058480.

[17] Whyte, J., Stasis, A. and Lindkvist, C. (2016) *Managing change in complex projects: configuration management, asset information and big data*. International Journal of Project Management, 34 (2). pp. 339-351. ISSN 0263-7863

[18] B. Ronen, T. Lechler and E. A. Stohr, "Xploring the Role of Production Management Concepts for Managing Projects: The "Divide and Conquer" Approach," *2017 Portland International Conference on Management of Engineering and Technology (PICMET)*, Portland, OR, 2017, pp. 1-7.
doi: 10.23919/PICMET.2017.8125352

[19] Cristóbal, José & Carral, Luis & Díaz, Emma & Fraguela, José & Iglesias, Gregorio. (2018). Complexity and Project Management: A General Overview. Complexity. 2018. 1-10. 10.1155/2018/4891286.

[20] Fitsilis, Panos & Damasiotis, Vyron & O’Kane, James. (2015). Scope management complexity in software projects. 10.13140/2.1.3992.7683.