Leveraging Transdisciplinary Engineering in a Changing and Connected World
P. Koomsap et al. (Eds.)

doi:10.3233/ATDE230616

Scope Attributes and Systemic Effect in Estimation Practices for Software Projects

Dipti GARG [a, 1] and Bryan R. MOSER [a, b]
[a] *Massachusetts Institute of Technology, Cambridge, USA*
[b] *University of Tokyo, Japan*

Abstract. Estimation in software projects allows organizations to predict scope, needed resources, and schedule leading to better performance. Inaccurate estimates may cause misalignment – amongst stakeholders and between the situation and design of the project – reducing performance and confidence in the development process. This paper focuses on software estimation, in particular attributes of scope and their usefulness to estimation. Scope is the tangible outcomes of project tasks. Attributes of scope include software functionality, dependence, and newness. A study of estimation practices began with semi-structured interviews conducted with leading software industry organizations. The interviewed practitioners characterized key steps involved in estimation and evaluated their organization's use of scope attributes. In addition, interview questions probed the consideration of scope topology and systemic effect. Findings from these interviews suggest that subjective assessment remains the most common method for estimation. Additionally, release level estimation processes are reported as informal and intuition-based, lacking analysis of systemic characteristics of scope especially dependence. Dependencies are often missing or insufficiently considered during estimation yet considered only by some at later stages of development. Findings from the interviews also suggest that estimation relies on priorities of perceived value rather than systemic criticality and project target feasibility. The next stage of this research proposes an expanded survey released to a wider practitioner population. Additionally, the findings suggest a need for longitudinal empirical studies to determine how knowledge of dependencies and analysis of systemic effect are imbued in organizations and become useful in estimation processes.

Keywords. Model-based Project Management, Scope Attributes, Software Estimation, Transdisciplinary Engineering Teams

Introduction

Scope is often considered the most important attribute for estimation since scope defines the boundaries and deliverables of a project. Brooks' classic work [1] emphasizes the non-linear combination in the interplay of scope, resources and time. Simply assuming that doubling the number of developers will halve the development time is often unrealistic. Many estimation methods (algorithmic and non-algorithmic) have been developed to formalize the estimation process. Most of the methods include some measure of software size as an input to the estimation process. Size been considered as

[1] Corresponding Author, Mail: dipti@mit.edu.

the an important factor that affects the estimation process [2]. Story Points, Function Points and Use Cases are amongst the other common metrics used to represent software size, especially in agile projects.

A review of related work reveals studies on estimation methods and their accuracy, yet few studies on the relationship between scope attributes (other than size) and their usefulness to the estimation method. The purpose of this paper is to understand the relationship between scope attributes and their usefulness to the entire estimation process.

1. Literature Review

Research and tools have been developed to address measures of completeness, quality, and size of project scope. [3] Some tools such as the Software Project Scope Rating Index (SPSRI) [4], though developed for evaluating scope definition, take into consideration other project elements such as resources and finances.

The complexity of scope is often considered at an aggregate level and characterized as an attribute of the overall project. In many studies, project complexity is a composite attribute of scope, resources, and process attributes. For example Griffin [5] discusses complexity and the impact on development time. Similarly, [6] Takai and Ishii, focused on overall project complexity as antecedent to new product development speed. These studies do not focus on scope item attributes such as number of functions or dependencies, but instead explored overall project characteristics to correlate with outcomes, especially development duration.

Systems analysis methods demonstrate that complexity is driven by elemental, pairwise, and topological complexity. For software projects the scope items are part of a project system, and thus not only the items themselves but also how they are coupled are significant [7]. Moser discussed the nature of dependence amongst scope items in modern projects, demonstrating that dependencies can be modelled more meaningfully than the classic representations of scope as sequential workflow [8][9].

Software estimate methods have been a research topic for quite some time [2], [10]–[14][15]–[18]. Some studies elaborate on different estimation methods and attempt to validate the methods' accuracy. However, most studies focused only on measuring the estimation methods' overall performance when testing them in an academic or industrial setting. It was surprising to see high error rates in accuracy found by some of these studies [19]. Moreover, studies do not clearly state which factors relate to the low accuracy of the estimation processes.

A majority of the estimation methods leverage software sizing metrics such as story points, use cases, software lines of code (SLOC) as input. However, the determination of these sizing metrics is often subjective. A more conclusive study would be beneficial to understand the project scope and how certain attributes influence the software size and thus, the usefulness of effort estimates.

Other researchers have explored the acceptance of these methods within industry. Formal / algorithmic methods such as COCOMO have not found acceptance within agile software communities, due to difference in philosophy and approach. For example – the original COCOMO method used Source Lines of Code (SLOC) as a key parameter in effort calculation. However, others focus on functionality delivered to the customer which does not necessarily correlate with the SLOC. For example – a highly efficient, experienced programmer can deliver the same functionality in fewer lines of code. Later

versions of COCOMO (COCOMO II) incorporated improvements to take in consideration the system-level functions and architecture.

2. Research Question

Software practices in recent decades have evolved quickly, with new tools, languages, methods, and distributed teamwork. Scope represents the ground truth of a project's deliverables and the most tangible attribute available for estimation; hence key to understanding the amount of effort required for a project. [8] Additionally, estimation is often based on drivers of size such as function points, use cases or agile user stories which require awareness of scope at the time of estimation.

Research is proposed to sample the most recent industrial practices for software project estimation and to detect any combined consideration of specific scope attributes and systems topology on the resulting scope size and thus software project estimations. In summary, an initial research question for this paper is:

RQ: How are software scope attributes represented for estimation methods across commonly accepted development frameworks?

To answer the research question, the following hypotheses were developed. The first hypothesis tests the relationship between frequently discussed scope attributes and the commonly used estimation methods. Based on the review of relevant literature, three key attributes pertaining were identified: Scope Functions, Dependencies, and Newness.

Scope Functions refer to the number of functions or use cases that need to be developed within the project's defined scope. Various methodologies, such as Use Case Point and Function Point Analysis, utilize the number of functions to assess the scope and its influence on the speed and complexity of development. Additionally, the size of the software is recognized as a significant factor in determining project complexity.

Scope Dependencies encompass the functional interdependencies among the elements within the project's scope. These dependencies contribute to the overall complexity of the project and can have direct or indirect effects on project cost and schedule

Scope Newness has been associated with the process, scope or technology newness. This paper defines newness as the degree to which the project requirements are similar to previous projects developed by the organization.

Despite being mentioned in the literature, other factors such as non-functional requirements, technology readiness, third-party components utilized, and lines of code are not considered in this work as their association with demanded scope items in project work are indirect (or in the case of outsourced scope, obvious).

The evidence from the literature is limited regarding sufficiency of scope attributes during estimation, especially dependencies. This leads to our first hypothesis as stated below.

> *H1: Currently accepted estimation methods in software projects do not sufficiently consider scope attributes, including number of functions, dependencies, and scope newness.*

The position of the scope element as a part of a software system can influence its criticality, thus impacting its significance in the estimation process. However, the

literature on the estimation process does not mention topology. This leads to a second hypothesis, which tests the consideration of typology within the estimation process.

H2: Current leading estimation method/s do not use topology to calculate emergent systemic effort.

3. Data Collection

A semi-structured interview method was selected to conduct interviews with real-world project practitioners. Interviews focused on US-based large software organizations with more than 3K employees and likely mature estimation processes. Interviews were conducted remotely via zoom to allow for recording and transcription.

A series of 11 interviews were conducted with professionals actively engaged in diverse areas of product development. These interviews focused on exploring the participants' firsthand experiences with estimation processes. The interviewees consisted of a group of 5 product managers, 2 engineers, 1 architect, and 3 program managers, all of whom were involved in the realm of software systems. The interviews involved experts from a total of 10 organizations, namely Appian Corporation, Dell Technologies, John Deere, Adobe, Amazon, Uber, Draper, PTC, Adobe, and CodeZero, collectively representing approximately 717 billion USD in revenue for year 2022.

The interviews were organized as follows. The first section was an open-ended discussion focusing on the context of the estimation process followed within the organization and challenges associated with the process. The second section was a series of specific then open-ended questions to gather insight on consideration of scope attributes and topology on the estimation process:

- In your recent experience of the estimation process, ***how well is the number of functions or use cases in a given project scope considered for the estimation purpose?*** Please rate on a scale of 1 to 5, where 1 is 'Not at all considered' and 5 is 'Considered to a great extent'. Please also explain why provided that rating.
- In your recent experience of the estimation process, ***how well are the scope dependencies taken into consideration for the estimation purpose***? Please rate on a scale of 1 to 5, where 1 is 'Not at all considered' and 5 is 'Considered to a great extent'. Please also explain why provided that rating.
- In your recent experience of the estimation process, ***how well is the scope newness taken into consideration for the purpose of estimation?*** Please rate on a scale of 1 to 5, where 1 is 'Not at all considered' and 5 is 'Considered to a great extent'. Please also explain why provided that rating.

To understand the impact of topology, open-ended questions were used to ensure common understanding of the question and to encourage further interpretation. Hence, a discussion approach yielded insights related to impact of scope topology on estimation:

- In your recent experience of the estimation process, have there been scope items that are considered because of their systemic impact on total system function, not necessarily driven by customer value or desirability?
- Is the position (connectedness) of scope in the overall system architecture taken into consideration for the purpose of estimation?

The transcribed interview data along with the scores from the closed-ended questions, were used to identify the patterns of scope elements, estimation methods and the relationship between them. The interview length varied between 40 to 60 minutes.

Consent for participation was obtained in advance of the interview and recorded verbally as part of the zoom recording.

4. Analysis

Each interviewee responded to three closed-ended questions on 'functions', 'dependencies' and 'newness' with a score between 1 (not at all considered) to 5 (considered to a great extent). Descriptive and inferential statistics were then calculated to identify significant differences or associations between the three metrics.

The three metrics had the same median (4), however 'dependencies' has a slightly smaller (3.8) mean than the other two. Additionally, 'dependencies' and 'newness' seem to have had a higher spread (standard deviation) than 'functions'. For 'functions', most of the participants provided a score of either 4 or 5.

Table 1. Summary statistics of three closed-ended questions.

	Dependencies	Functions	Newness
Mean	3.8	4.3	4.2
Std. Dev.	1	0.6	1
Median	4	4	4

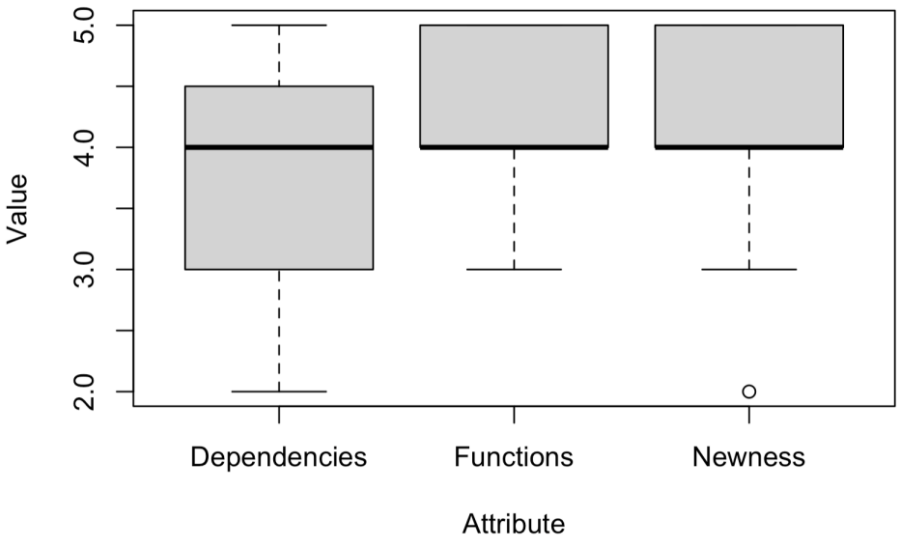

Figure 1. Boxplot analysis of responses to three close-ended questions. Y-axis shows the range of values (from 1 to 5, with 1 as not at all considered and 5 as considered to a great extent) for the three close-ended questions and the position of their median value.

One prominent pattern that emerges relates to *dependencies*. The pattern indicates that consideration of dependencies is generally weaker than the other attributes. Additionally, the boxplot and histogram demonstrated a higher variability within dependencies relative to *functions* or *newness*. The higher spread within *dependencies*

suggested greater range of consideration in estimation relative to the others. This was also observed as part of the unstructured data analysis.

- 'Newness' and 'Functions' illustrate a ceiling effect with responses being clustered toward the higher end of the measurement.

- Correlation analysis was performed between 'functions', 'dependencies' and 'newness'. However, with 95% confidence, none of the pair-wise associations were found to be significant.

5. Affinity Analysis and Emergent Themes

Affinity analysis [20][21] was applied to disclose groupings of similar thoughts among the narrative comments. The following steps were performed to identify themes –

1. Interview verbatims were gathered and transcribed using auto-transcription feature available in zoom.
2. Recurring topics, ideas, or concepts emerging from the interview verbatims were identified.
3. Relevant verbatim excerpts representing each identified theme were extracted. These excerpts captured the essence of the participants' responses related to the specific theme.
4. The extracted verbatim excerpts were reviewed and grouped based on their similarity or shared meaning. Commonalities, patterns, or connections between the excerpts were identified.
5. Affinity groups were created from the grouped verbatim excerpts. These labels reflected the overarching themes or topics that emerged from the interviews.
6. The affinity groups were arranged in a logical and meaningful way.

Five affinity groups were found in the comments, and these became the headlines for the findings. From the analysis of the relationships between the groups and their contribution to the overall research topic, the overriding theme which emerges from the interviews:

Estimation processes at release level are informal and intuition based, lacking a systematic analysis of scope attributes, especially dependencies.

#1 T-shirt size estimation is commonly used for release planning. High-level estimates of 'T-shirt' size are used for release planning, with only prioritized requirements included in the planning process. Prioritization is significantly based on customer value and does not consider system topology. There were no mentions of systemic or topological significance in the interviews conducted.

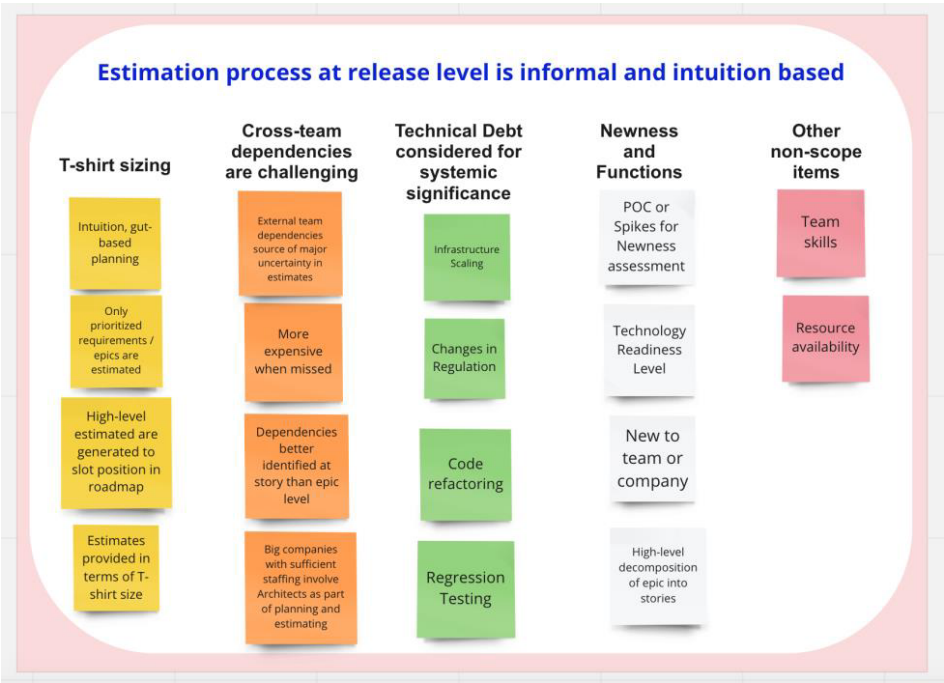

Figure 2. Affinity analysis based on interview transcripts.

#2 Cross-team scope dependencies represent a challenge for estimation. Most practitioners interviewed mentioned dependency identification and consideration as important areas of improvement for their estimation process. There was a concern about how missed dependencies led to schedule delays, poor customer experiences, and rework. When missed, they were found to be much more expensive than other scope attributes. Furthermore, the practitioners who mentioned that they considered dependencies significantly, did not mention any formal ways of representing and analyzing project dependencies. Additionally, it was mentioned by a few participants that they better identified the dependencies at an individual story-level as by that time, more detailed requirements are available.

#3 Technical Debt related scope is considered for systemic significance. Scope items related to technical debt such as infrastructure scaling or regulation changes were the only items considered for their significance to systemic criticality. However, none of the participants mentioned any systematic way of assessing the criticality of these items other than the perceived importance. Two participants mentioned that the sprint buffer capacity would be used to accommodate such items; thus, these would always be released along-with an item related to customer-value.

#4 Newness and Functions are consistently considered scope attributes. Scope Newness and Functions were consistently mentioned as significant contributors to the estimation process. The assessment of functions was based on the number of features or stories derived from a given scope. Newness was mostly measured in-terms of new to a given team or new to the organization based on prior projects undertaken by the organization. It referred to the product technology novelty (and the process novelty) as differentiated in the research by [22]. Except one, none of the participants considered

Newness in terms of 'Technology Readiness Level'. Most of the participants mentioned a process of POC (Proof of Concept) or Spike to evaluate feasibility of a highly new scope, before it is included in an estimation process.

#5 Resource availability and skills were non-scope related items significantly considered for estimation. Internal resource availability and team skills were the other significant considerations during estimation. Certain scope items had specific resource requirements that would highly impact the estimate.

6. Hypothesis Testing

Based on the practitioner interviews and affinity analysis, a test of the hypotheses was considered. These preliminary insights will guide the refinement of the hypotheses and improvement of the survey for the next phase of this research.

H1: Currently accepted estimation methods in software projects do not sufficiently consider scope attributes, including number of functions, dependencies, and scope newness. The closed-ended questions suggest that while scope functions and newness were always considered, limited evidence for scope dependence was shown. Additionally, the affinity analysis suggests that release-level estimation is informal and intuition-driven with no systematic way of analyzing the attributes, especially dependence. The estimates are an assessment at that point with what is known locally at the time, which may be limited. The estimation process is reported to be more accurate at the level of sprint planning since the scope is fully broken into detailed stories. However, key decisions related to roadmap release planning and resource scheduling are already made by that point. Hence, based on the interviews conducted, there isn't sufficient evidence that scope attributes are sufficiently considered during the estimation process.

H2: Current leading estimation method/s do not use topology to calculate emergent systemic effort. Based on the unstructured interviews, there was no mention of system topology during the estimation process. Each scope item (epic or feature) was assessed individually with the total release effort calculated as the sum of the effort of individual scope items. Participants, however, mentioned infrastructure related items when asked for non-customer value related items. These items, even though important, seemed to be considered because of software sustainability and scalability aspects and not necessarily because of their topological effect on other scope items. Furthermore, it appears that agile estimators have a short-term planning focus with emphasis only on prioritized scope items. The process of sprints with focus on backlog burndown seems to work for them. One reason reported by the practitioners is their bias for action over planning and belief that changes are easy to incorporate. Based on the interviews, there is little evidence that current leading estimation methods use topology to calculate emergent systemic effort.

7. Conclusion

This paper presents a preliminary study to understand current estimation methods used in well-established software development teams. While all the organizations interviewed consider scope attributes, evidence was not found revealing robust scope analysis performed, especially use of topology to understand systemic effect and criticality. Based

on the interviews, expert judgment was found to be the most common method of estimation.

The survey identified scope dependence as less represented than other scope attributes (functions and newness). Estimates appear to be temporally short sighted – reaction to what is known locally and at the time. Scope dependencies are often identified much later in a project, and thus consideration is limited or missed. Furthermore, participants expressed that missed dependencies are a major source of estimation error and are expensive when missed.

The motivation of this paper was to understand how scope characteristics are useful to transdisciplinary engineering teams in assessing complexity and size during software estimation. While the literature review helped identify some characteristics, interviews were conducted to understand how these attributes are used in practice. Additionally, related literature describes estimation methods, especially more quantitative, parametric methods. However, none of the interviewees leveraged a sophisticated estimation method other than expert opinion and subjective judgment. It is not year clear if the lack of sophisticated methods was due to limited time during the estimation process, unavailability of the required information needed, or lack of skills/knowledge to perform such analyses.

The practitioners in this study did report use of scope attributes, especially functions and newness, yet without concrete evidence of formal methods to represent and assess these attributes. This absence of formalized process raises doubts on the credibility of the estimation process. One might expect an intuition-driven process to be more prevalent in smaller, start-up companies, with few resources and time to conduct extensive estimation. However, a surprise in this study revealed that large organizations' estimation process do not differ as expected. Sophisticated methods used to estimate larger projects were not evident. However, even with estimation as an informal process, estimation did promote trade-offs and insight for scheduling projects and feature requests in a roadmap.

Findings from the interviews suggest that release planning and estimation mainly consider prioritized scope items, and thus may be inadequate to perform topological assessment of scope due to dependence. Scope prioritization in each release was shown to be based on customer value rather than topological criticality. However, technical debt, infrastructure scaling, and regulation-related changes were items considered based on topological importance. While such systems assessment can promote a view of interconnections, the methods may be resource-intensive and not shown as common in software organizations. Organizations with a designated architect role may have a more active input in understanding system topology and developing a better understanding of cross-team dependencies. However, only a few interviewees mentioned the involvement of architects during estimation to understand the impact of topology. Most recognized missed dependencies to be a challenge during estimation. This challenge demonstrates that there is an opportunity for practitioners to consider system topology as part of the estimation process, which may help in surfacing obscure dependencies.

The gaps shown in the interviews suggest that organizations should better utilize mechanisms to identify dependencies up-front and include them in the estimation process. Early identification of dependencies would reduce cross-team communication efforts and emergency work giving all the teams a better chance to estimate and plan. Practitioners may also consider the full scope (and not only the customer-driven scope items) to determine the topological complexity of the system and its effect on integration effort.

Finally, further investigation of dependencies needs to be performed to understand methods used by organizations to estimate systemic dependence. Future research will involve developing a more structured survey to understand the identification, representation, and consideration of dependencies during the estimation process.

References

[1] F.P. Brooks, *The Mythical Man-month*. Addison-Wesley Pub. Co, Reading, 1975.
[2] F.J. Heemstra, Software cost estimation, *Inf. Softw. Technol.* , 1992, vol. 34, no. 10, pp. 627–639.
[3] S. Amjad *et al.*, Calculating Completeness of Agile Scope in Scaled Agile Development, *IEEE Access*, 2018, Vol. 6, pp. 5822–5847
[4] I.U. Hassan and S. Asghar, A Framework of Software Project Scope Definition Elements: An ISM-DEMATEL Approach, *IEEE Access*, 2021, Vol. 9, pp. 26839–26870.
[5] A. Griffin, The Effect of Project and Process Characteristics on Product Development Cycle Time, *J. Mark. Res.*, 1997, vol. 34, no. 1, pp. 24–35.
[6] J. Chen, F. Damanpour, and R. R. Reilly, Understanding antecedents of new product development speed: A meta-analysis, *J. Oper. Manag.* , 2010, Vol. 28, no. 1, pp. 17–33.
[7] K. Sinha and O. L. de Weck, Empirical validation of structural complexity metric and complexity management for engineering systems, *Syst. Eng.*, 2016, Vol. 19, No. 3, pp. 193–206.
[8] B. Moser, Scope Patterns for Projects Modelled as Sociotechnical Systems, *The Journal of Modern Project Management*, January-April 2017, pp. 118–122.
[9] B. Moser, W. Grossmann, and P. Starke, Mechanisms of dependence in engineering projects as sociotechnical systems, *Advances in Transdisciplinary Engineering*, 2015, Vol. 2, pp. 142–151.
[10] H. Leung and Z. Fan, Software cost estimation, in *Handbook of Software Engineering and Knowledge Engineering: Volume II: Emerging Technologies*, World Scientific, 2002, pp. 307–324.
[11] G. Karner, Resource estimation for objectory projects, *Object. Syst. SF AB*, 1993, vol. 17, no. 1, p. 9.
[12] E. Coelho and A. Basu, Effort estimation in agile software development using story points, *Int. J. Appl. Inf. Syst. IJAIS*, 2012, vol. 3, no. 7, pp. 7-10.
[13] V. Mahnič and T. Hovelja, On using planning poker for estimating user stories, *J. Syst. Softw.* , 2012, Vol. 85, No. 9, pp. 2086–2095.
[14] M. Usman, E. Mendes, F. Weidt, and R. Britto, Effort estimation in agile software development: a systematic literature review, *Proceedings of the 10th international conference on predictive models in software engineering*, 2014, pp. 82–91.
[15] S. Kusumoto, F. Matukawa, K. Inoue, S. Hanabusa, and Y. Maegawa, Estimating effort by use case points: method, tool and case study, *10th Int. Symposium on Software Metrics*, 2004. pp. 292–299.
[16] B. Anda, H.C. Benestad, and S. E. Hove, A multiple-case study of software effort estimation based on use case points, *International Symposium on Empirical Software Engineering*, 2005, pp. 407-416.
[17] S. Grimstad and M. Jørgensen, Inconsistency of expert judgment-based estimates of software development effort, *J. Syst. Softw.* , 2007, Vol. 80, No. 11, pp. 1770–1777.
[18] S. Grimstad and M. Jørgensen, Preliminary study of sequence effects in judgment-based software development work-effort estimation, *IET Softw.* , 2009, vol. 3, no. 5, pp. 435–441.
[19] C. F. Kemerer, An empirical validation of software cost estimation models, *Commun. ACM*, 1987, Vol. 30, no. 5, pp. 416–429.
[20] C. Remy, G. Harboe, J. Frich, M. M. Biskjaer, and P. Dalsgaard, Challenges and Opportunities in the Design of Digital Distributed Affinity Diagramming Tools, *Proceedings of the 32nd European Conference on Cognitive Ergonomics*, 2021, pp. 1–5.
[21] J. Liu and J. Eagan, ADQDA: a cross-device affinity diagramming tool for fluid and holistic qualitative data analysis, *Proc. ACM Hum.-Comput. Interact.* , 2021, Vol. 5, no. ISS, pp. 1–19.
[22] M. V. Tatikonda and S. R. Rosenthal, Technology novelty, project complexity, and product development project execution success, *IEEE Trans. Eng. Manag.*, 2000, Vol. 47, no. 1, pp. 74–87.