An Analysis of Students' Teams Scores at the 2022 Software Estimation Challenge

Donatien Koulla Moulla [1,2], Jean-Marc Desharnais [3] and Alain Abran [3]

[1] *University of South Africa, The Science Campus, Florida, 1710, South Africa*
[2] *University of Maroua, Maroua, P.O. Box 46, Cameroun*
[3] *École de Technologie Supérieure, 1100, rue Notre-Dame Ouest, Montréal, Québec, H3C 1K3, Canada*

Abstract

This paper presents an analysis of the 2022 edition of the 'Software Estimation Challenge' organized by the COSMIC Group. The challenge is based on best practices in software effort estimation, including the use of the COSMIC – ISO 19761 standard for sizing software requirements and the early sizing of software functional and non-functional requirements allocated to software functions. The three major components of this challenge consist of sizing the software requirements of a case study, developing an estimation model, and using it to estimate the development effort for the case study provided. While a previous study was based on a survey of a sub-group of 22 teams who participated in the 2022 edition of this challenge, the study reported here is based on the analysis of students' team scores across teams and contexts of participation. To help the teams' tutors and students plan and prepare for future challenges, this study presents an analysis of how teams performed across each of the challenge tasks. In summary, the teams performed best in the tasks limited to the application of preprogrammed statistical formula, and much worse in tasks requiring analytic skills for the sizing of the requirements or in the application of their reasonably well-built estimation model to the practical case study they had sized (that is, moving from theory to practice).

Keywords

Software competition, software estimation, COSMIC, ISO 19761, function points, gamification

1. Introduction

Software project estimation is important for allocating resources and planning reasonable work schedules. Although there is a large body of literature on software estimation, estimating software projects remains a challenging activity, and software managers still struggle to deliver their projects according to the estimated budget, deadlines, and expected functionalities.

Identifying the best practices in software estimation and teaching them to students who are often more interested in technical tasks than management tasks is challenging. There are also difficulties in teaching software estimation when it comes to moving from theory to practice [1], and there is typically a lack of case studies on learning purposes and acquiring estimation skills [2].

To enable students to make this link, some approaches have been proposed, such as in [3], with data for educational purposes, covering various estimation topics with related questions at the end of each chapter. A more proactive approach is Humphrey's Personal Software process (PSP) and Team Software Process (TSP) to assist engineers in developing their own individual software engineering skills, including estimation by coaching and guidance [4]. Other studies have proposed gamification as a better alternative for teaching software engineering estimation [5, 6, 7]. This approach creates a competitive and collaborative environment for learners to improve their interest, knowledge, and skills

IWSM-Mensura, September 14–15, 2023, Rome, Italy
EMAIL: moulldk@unisa.ac.za; jean-marc.desharnais@etsmtl.net; alain.abran@etsmtl.ca
ORCID: [orcid.org/0000-0001-6594-8378]

in the application of software estimation best practices. For instance, the SIMSE (SIMulation of Software Environment) software game developed by Navarro [8] and the ‘Easy Estimation with Story Points’ game developed by Agile Learning Labs [9]. Similarly, Marin et al. in 'An adventure Serious Game for Teaching Effort Estimation in Software Engineering' [10] designed and developed the “Back to Penelope” game based on the COSMIC sizing method to make effort estimation more attractive for students: this game consists of, among other tasks, sizing software functions from the class diagrams. Appendix A provides an overview of the COSMIC sizing method.

The Ouhbi et al. survey [11] of software engineering faculty and experts also reported that students’ engagement is a great concern, and to address the issue, new teaching methodologies are needed, including gamification and problem-based learning. However, the implementation of a game-based learning approach is challenging for instructors [12].

To increase students’ awareness of the importance of software estimation and the necessity of acquiring the estimation knowledge and skills required, the COSMIC Group[2] was designed and implemented in 2020 as an international software estimation challenge in which university students compete in a team mode. This challenge addresses the concerns discussed in [11] to address students’ engagement, gamification, problem-based learning, and easing instructors’ challenges in [12]. This estimation challenge is designed based on best practices in software effort estimation, including the use of the COSMIC – ISO 19761 standard for sizing software requirements, which consists of determining the size of the software to be developed and described in a case study, developing an estimation model using a given set of historical data, and estimating the effort to develop both the given functional and non-functional requirements.

The aim of this challenge is to provide students with team-based learning experience in software effort estimation. Furthermore, the study material recommended by the COSMIC Group on some estimation best practices, as well as challenge preparatory material provided on the web by the COSMIC Group, guide university teachers-mentors in structuring and delivering the necessary teaching material to prepare their participating students’ teams for the challenge, either as an elective or mandatory course activity.

In the 2022 edition of this challenge, 53 teams participated, for a total of 233 students from five countries. An initial study [13], based on a survey of a sub-group of 22 teams who participated in the 2022 edition of this challenge, reported that in students’ opinions, this kind of competition experience was helpful in learning the COSMIC software sizing method and improving their own knowledge and skills in software estimation. The study reported here differs from [13] in the following way: it is based on the analysis of students’ team scores, rather than on opinions, and analyzes these scores across teams and contexts of participation. Of course, both types of study are useful and complementary.

This paper presents the design of this challenge in terms of the estimation tasks to be carried out by students as well as an analysis of how teams from various contexts performed overall and across each of the challenge tasks. The findings reported here can provide future challenges to participants with additional insights into the knowledge acquired prior to the challenge. These findings can also help teams’ mentors to improve their software estimation teaching strategies and related training materials.

The remainder of this paper is organized as follows. Section 2 presents an overview of the estimation approach for this challenge, including the COSMIC software-sizing method. Section 3 presents the students’ team scores in the 2022 challenge, as well as the strengths and weaknesses identified by the estimation tasks. Section 4 presents the discussion and insights learned.

2. Challenge Design

2.1. Challenge structure

This Estimation Challenge was designed as a three-phase process (Figure 1):

Phase A - Sizing of the case study.

[2] COSMIC: Common Software Measurement International Consortium - a non for profit organization responsible for maintaining the COSMIC – ISO 19761 standard for functional size measurement - see https://cosmic-sizing.org

The challenge input is a Case Study describing a number of systems and software requirements, some with detailed information and others at a high-level only. This phase consisted of four tasks for sizing with COSMIC – ISO 19761 of the software requirements provided in the case study:

- Task 1 – Precise sizing of the functional requirements described at a detailed level.
- Task 2 – Approximate sizing of functional requirements available only at a high level of description.
- Task 3: Sizing of the system non-functional requirements allocated to software functions.
- Task 4 – Calculation of the total COSMIC size of the case study.

Phase B - Building an estimation model.

The challenge inputs for Phase B are a dataset of project data in an Excel format, including size, effort, etc. Phase B consisted of two tasks.

- Task 5: Constructing a linear regression estimation model using the dataset provided.
- Task 6: Documenting the errors intervals of the estimation model.

Phase C- Effort estimation for the case study.

The challenge inputs for Phase 3 are the outputs of Phases A and B as well as a specified development environment. This phase consists of the following two tasks:

- Task 7: Estimate the development effort for the case study size in Phase 1 using the estimation model developed in Phase B.
- Tasks 8: Convey the estimations outcomes to management in a PowerPoint format.

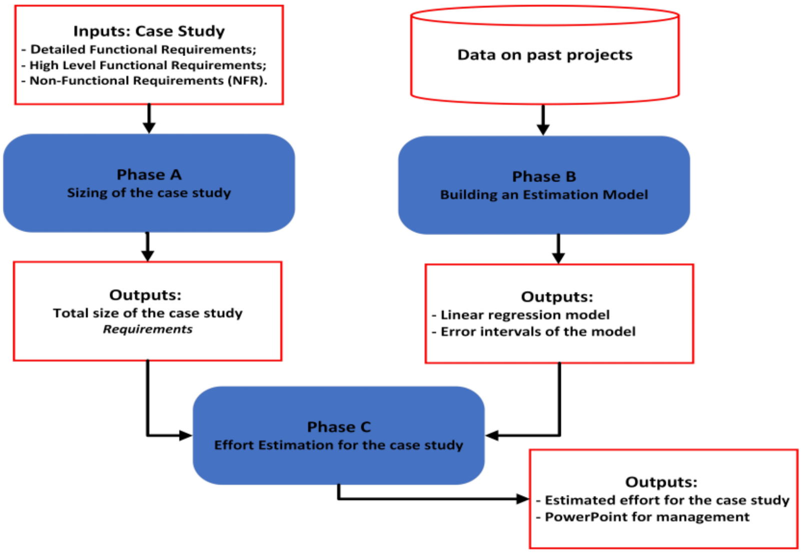
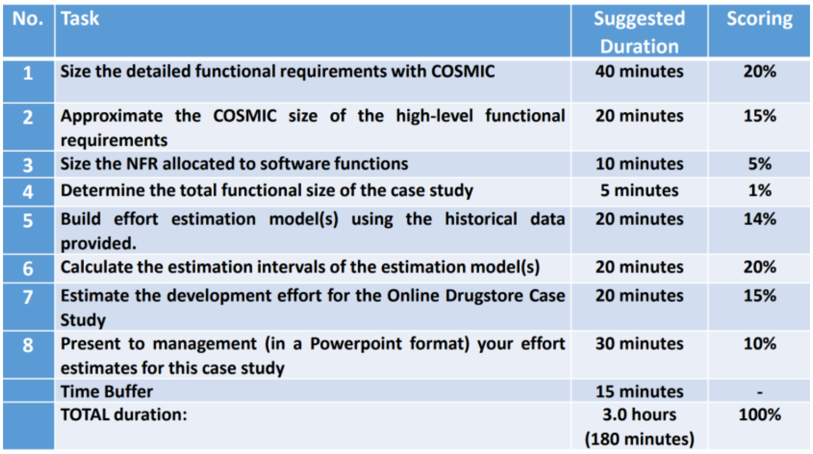

Inputs: Case Study
- Detailed Functional Requirements;
- High Level Functional Requirements;
- Non-Functional Requirements (NFR).
Data on past projects
Phase A
Sizing of the case study
Phase B
Building an Estimation Model
Outputs:
Total size of the case study
Requirements
Outputs:
- Linear regression model
- Error intervals of the model
Phase C
Effort Estimation for the case study
Outputs:
- Estimated effort for the case study
- PowerPoint for management

Figure 1: Software Estimation Challenge Process in 3 Phases

Figure 2 presents suggested durations per task and their respective scoring percentages for a total maximum of 3 hours for the challenge and a maximum score of 100 % for all tasks.



No.	Task	Suggested Duration	Scoring
1	Size the detailed functional requirements with COSMIC	40 minutes	20%
2	Approximate the COSMIC size of the high-level functional requirements	20 minutes	15%
3	Size the NFR allocated to software functions	10 minutes	5%
4	Determine the total functional size of the case study	5 minutes	1%
5	Build effort estimation model(s) using the historical data provided.	20 minutes	14%
6	Calculate the estimation intervals of the estimation model(s)	20 minutes	20%
7	Estimate the development effort for the Online Drugstore Case Study	20 minutes	15%
8	Present to management (in a Powerpoint format) your effort estimates for this case study	30 minutes	10%
	Time Buffer	15 minutes	-
	TOTAL duration:	3.0 hours (180 minutes)	100%

Figure 2: 2022 Estimation challenge: tasks, duration and scoring percentages

2.2. Recommended resources in preparation for the challenge

- Phase A: For sizing the requirements, the following documents, available free on the web, are recommended:
    - COSMIC Measurement Manual for ISO 19761 Part 1: Principles, Definitions and Rules, https://cosmic-sizing.org/publications/measurement-manual-v5-0-may-2020-part-1-principles-definitions-rules/;
    - COSMIC Measurement Manual for ISO 19761 Part 2: Guidelines, https://cosmic-sizing.org/publications/measurement-manual-v5-0-may-2020-part-2-guidelines/;
    - Course Registration (‘C-REG’) System Case Study, v2.0.1, https://cosmic-sizing.org/publications/course-registration-c-reg-system-case-study-v2-0-1/;
    - Software Project Estimation book (Chapters 5, 6 & 9) [14];
    - Early Software Sizing with COSMIC: Practitioners Guide, https://cosmic-sizing.org/publications/early-software-sizing-with-cosmic-practitioners-guide/;
    - Non-Functional Requirements and COSMIC Sizing Practitioner’s Guide, https://cosmic-sizing.org/publications/non-functional-requirements-and-cosmic-sizing-practitioners-guide/.

  These resources provide definitions, rules, and guidelines for the COSMIC method, including many examples and some of the best practices in software project estimation.

- Phase B: To develop an estimation model and identify related error intervals, any textbook on linear regression models is recommended, including Chapters 5, 6, and 9 of [14].

Of course, professors-mentors can recommend to their students whatever additional material they deem relevant in preparation for the challenge.

3. Teams’ Performance in the 2022 Estimation Challenge

An estimation challenge, by design, is not expected to be easy:

- On one hand, a challenge provides an opportunity for all students to learn new skills and demonstrate their skills in applying them through a case study.
- On the other hand, it should allow the identification of the students’ teams that perform best in software estimation.

This section presents the teams’ performance in terms of the total scores of the teams, followed by the scores per task and contexts of participation.

3.1. Teams' overall challenge performance

The distribution of the overall challenge scores is presented in Table 1, with intervals of 10 points from the highest to the lowest, with an average of 56% and a median of 61% for all teams.

- The three winning teams from two distinct universities achieved scores over 80%, clearly above all other teams.
- 25 out of 53 teams achieved scores between 60 and 79%, doing reasonably well in this challenge. Together with the three winning teams, 64% of the teams scored higher than 60%.
- The Teams with some scores lower than 15% did not perform some of the tasks.

Table 1 Challenge scores distribution across all teams

Distribution	Number
90-100	1
80-89	2
70-79	12
60-69	13
50-59	8
40-49	4
30-39	8
20-29	2
10-19	1
0-9	2
Total	**53 teams**

Figure 3 presents the histogram of the score distribution across all teams and, in the blue, what would have been a Gaussian distribution? There is no *a priori* reason to expect a Gaussian distribution (e.g., a normal distribution) in a learning context. However, a comparison with the normal distribution the right skew in Figure 3 indicates that a large number of teams performed fairly well in the challenge and that the challenge overall was not easy, with only a few teams achieving a score higher than 80%.

Furthermore, in a typical student population, not all students study all of the teaching materials well, and a number of low scores are expected; for instance, the study in [19] reported that out of the 56 students from 22 teams who participated in the survey of this 2022 estimation challenge:

- 9% of the students did not attend the lecture in preparation for the challenge;
- 20% spent less than an hour in preparation for the challenge;
- 30 % spent between 1 and 3 hours in preparation, and
- 50% spent between 4 to 6 hours of preparation.

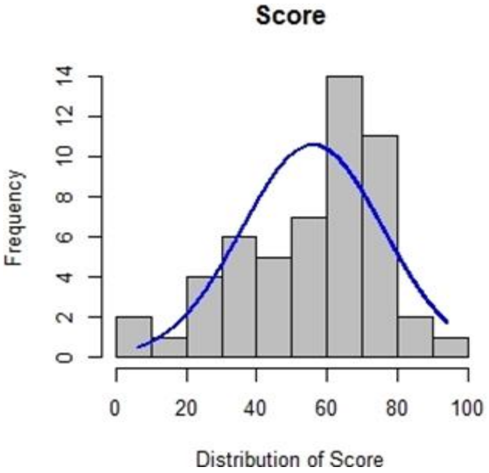


Figure 3: Distribution of teams scores (and Gaussian Distribution in bleu)

3.2. Teams' performance per task

This section presents an analysis of the teams' performances per task, including the 53 teams average, median, and percentage of scores per task (see Table 2). In Table 2, tasks 3 and 4 are combined, and task 4 consists of a simple addition of the size of the nonfunctional requirements to the size of the functional requirements.

Table 2 Scores per task across all teams

	Task 1	Task 2	Tasks 3-4	Task 5	Task 6	Task 7	Task 8
Max score per task	20	10	6	14	20	15	15
Team average	9.8	5.2	3.0	9.7	15.5	6.8	6.0
Team Median	11	5	2	8	20	5	6
Team relative average score	49%	51%	50%	69%	78%	45%	40%

From Table 2 and Figure 4, it can be observed that the teams' average scores were:

- the best (e.g., over 60%) of the two tasks limited to the application of pre-programmed statistical formula to the provided set of project data:
    - task 5: Building an effort estimation model using the linear regression technique,
    - Task 6: Calculating the estimation intervals in terms of Mean Magnitude of Relative Error (MMRE) and coefficients of determination (R^2).
- lower (below 51%) in all other tasks requiring analytic skills for the sizing of the requirements or in the skills at the mapping of their reasonably well-built estimation model to the practical case study (that is, moving from theory to practice).

Figure 4 shows a major left skew with very low average scores in the following tasks, indicating that a large number of teams had either not been aware of the underlying concepts or had not figured out how to apply them in the practical situation presented in the case study:

- Tasks 2: application of early sizing techniques to high level requirements.
- Tasks 3-4: application of sizing of nonfunctional requirements allocated to software functions.
- Task 7: application in practice of the mathematical models to the case study they had sized.
- Task 8: presentation to management of key findings using the mandatory PowerPoint approach of communication (note that a few teams did not prepare such a PowerPoint, indicating that the information was already available in the details of the previous task, without acknowledging that 'information has to be communicated').

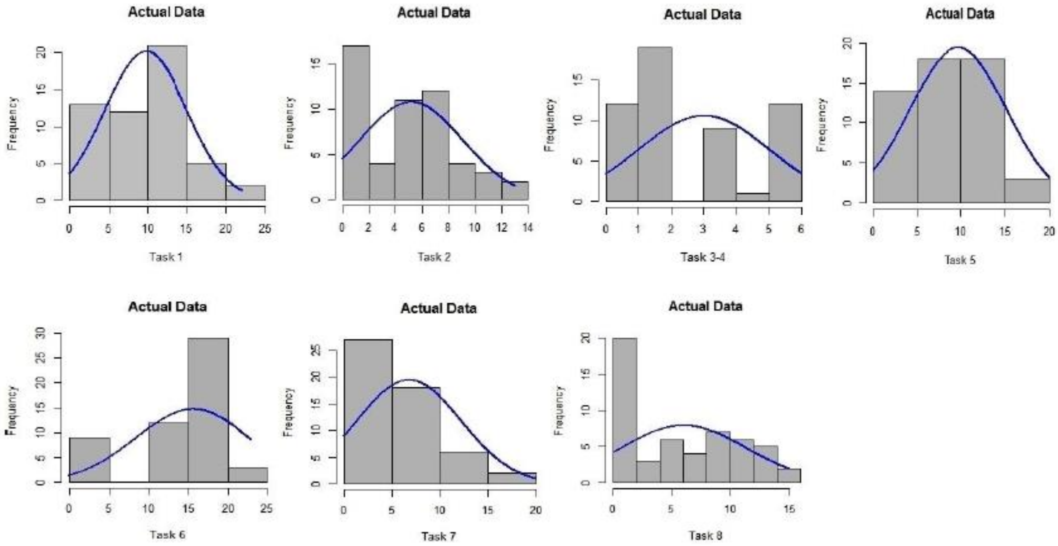


Figure 4: Histogram of estimation tasks scores

3.3. Teams' performance by participation context

3.3.1. Participation contexts

The participation context of the 53 teams is characterized in the following way:

A. An elective context outside the course curriculum: Within this context, the students did not receive any formal academic recognition. The seven teams within this 'elective' context were mostly master's degree students from five countries: Cameroon, Canada, Egypt, Mexico, and Turkey.

B. Mandatory academic activity within a scheduled course: In this context, students received formal academic recognition, a portion of which was based on their teams' overall score at the challenge. The students within this 'mandatory' context were all bachelor's degree students who were formally registered in a software project management course. This group can be further distinguished in the following way:

B1. From a single course at a Turkish university, co-taught by two teachers as a single teaching group, the students split themselves into 22 teams. This was the first university to participate in this challenge.

B2. From a single course at a Canadian university, where students were registered in two teaching groups with two distinct teachers using the same course content and set of exercises, together with a single lab instructor for both groups: these students split themselves into 25 teams. This was the second participation of this university with the same teachers and lab instructor in charge of tutoring the students in preparation for the challenge for both years of participation.

3.3.2. Average score per context

To analyze whether these three contexts led to differences in team performance, the full dataset was split into corresponding subsets. Figure 5 presents the average (in blue) and median (in red) scores by the participation context.

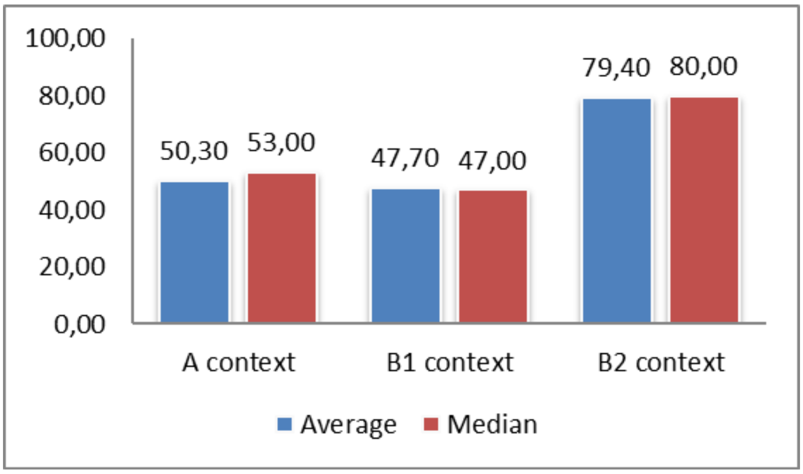


Figure 5: Average and median scores across elective and mandatory participation contexts

From Figure 5:

- The teams in their first year of participation (i.e., A and B1 contexts at the master and bachelor levels, respectively) achieved similar scores.
- The teams from the university with two years of participation (e.g., the B2 context at the bachelor's level) achieved the highest scores in comparison to the two other contexts.

In Figure 5, the average and median values are similar within each participation context.
Distribution of scores across contexts

Histograms of the distribution of scores across the contexts are presented in Figure 6. It should be noted that on the horizontal axis, the range of scores varies across contexts, with the following observations:

- The maximum score was 70% in the elective context A.
- The maximum score is higher than 100% in context B1 and B2 since a few teams got some ‘bonus’ points for additional relevant estimation information provided by best teams.

In terms of the distribution of the team scores across contexts, it can be observed that:

- for the A and B1 contexts of first year of participation, many teams have scores lower than 50%, none above 70% for the A context, and few for the B1 context.
- for the B2 context of the 2nd year of participation: there is a single team with a very low score, while all others have a score higher than 60%. It should be noted that, in this context, the team mentors improved their teaching and mentoring based on lessons learned in the previous participation, which benefited the students.

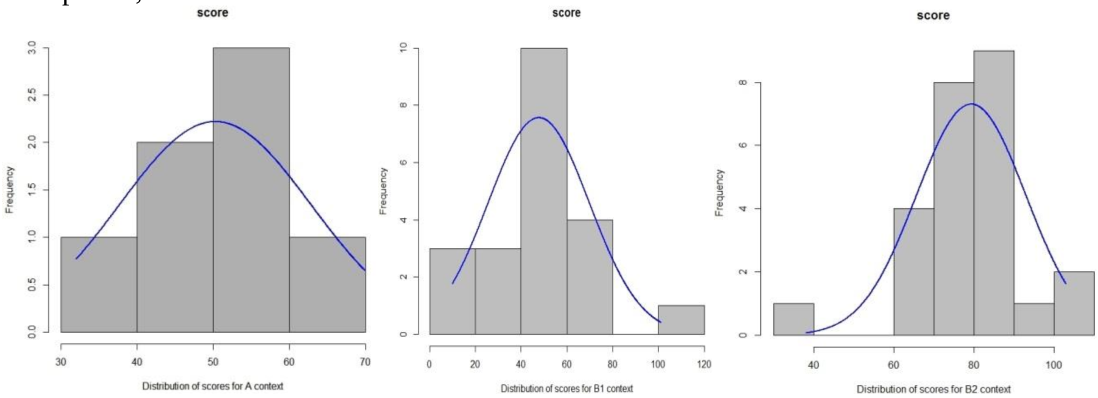


Figure 6: Histogram of scores distribution per participation context

3.4. Teams’ performance by team sizes

Table 3 presents the team size in terms of the number of participants in a team (from 2 to 7), number of teams per team size, their average, and median overall performance scores.

Table 3

Average and median scores per teams’ size

Teams size (number of team members)	Number of teams	Average Score	Median Score
2	5	6.00	2.00
3	20	6.75	5.00
4	5	8.23	6.00
5	6	8.93	7.50
6	6	10.00	10.00
7	11	9.43	10.00

From Table 3, it can be observed that the teams’ performance in terms of their average score improved with the number of additional participants. It was also observed that the average team size for contexts A and B1 was three students, whereas the average team size for B2 context was twice as large for six students per team.

4. Discussion and Insights Learned

The study reported here is based on the analysis of students’ team scores across teams and contexts of participation. The students who participated in the challenge were from different countries, the size of the teams varied, and the participating teams received different training in software requirement sizing and estimation tasks. Teaching strategies, as well as students' motivations and abilities, also influence their knowledge acquisition and skills development.

For the analysis of the teams' performance, they were classified into three main contexts of participation: A (Elective), B1 (Mandatory – 1st year of participation), and B2 (Mandatory – 2nd year of participation). For the overall performance on this challenge, teams from the B2 context came from an environment with previous experience in the challenge and having developed some prior 'institutional knowledge': same teachers and instructor, but with different students and students' teams. To facilitate fair play in the 2022 edition of this challenge, these teachers and instructor volunteered through the COSMIC Group to share publicly with all 2022 registered teams an improved version of the initial training material they had developed for their first year of participation.

While a second year of institutional participation with additional expertise in mentoring participating teams may be posited to provide an edge in terms of improved performance, it may well be argued that the larger teams provided such an edge for their higher performance.

In summary, the teams performed:

- best (e.g., over 60%) in the two tasks limited to the application of preprogrammed statistical formula to the provided data set of project data:
    - task 5: Building an effort estimation model using a regression, and
    - task 6: Calculating the estimation intervals.
- much worse (below 51%) in tasks requiring analytic skills for the sizing of the requirements or in the skills at the mapping of their reasonably well-built estimation model to the practical case study the teams had sized (that is, moving from theory to practice).
- Some of the teams might not have been mentored-trained into the most recent best practices developed by the COSMIC Group and included in this challenge, such as in the early sizing techniques of software requirements with few details, as well as in the sizing of system non-functional requirements allocated to software functions.

The findings reported here can provide future challenges participants with additional insights into the knowledge acquired prior to the challenge. These findings can also help teams' mentors to improve their software estimation teaching strategies and related training materials. In future work, we plan to investigate in a much deeper way the quantitative data we gathered. Moreover, we plan to make further analyses by taking into account contextual information per team/student (e.g., process compliance/understanding, knowledge/background of students). We also plan to include a more in-depth discussion about the distinctive results of the B2 teams.

5. Acknowledgment

We are grateful to the anonymous reviewers.

6. References

[1] A. Dagnino, Increasing the effectiveness of teaching software engineering: A University and industry partnership, in: Proceedings of the 27th Conference on Software Engineering Education and Training (CSEE&T), IEEE, Klagenfurt, Austria, 2014, pp. 49-54. doi: 10.1109/CSEET.2014.6816781.

[2] N.C. Flores, M.A. Muñoz and J.G. Hernández Reveles, Guide to teach Lean Startup methodology to software engineering students using a serious game, in: Proceedings of the Mexican International Conference on Computer Science (ENC), IEEE, Morelia, Mexico, 2021, pp. 1-8. doi: 10.1109/ENC53357.2021.9534827.

[3] B. Boehm, Software Engineering Economics. In: Broy, M., Denert, E. (eds) Pioneers and their Contributions to Software Engineering. Springer, Berlin, Heidelberg, 2001.

[4] W.S. Humphrey, PSP(SM): A Self-Improvement Process for Software Engineers, Addison-Wesley Professional, Upper Saddle River, 2005.

[5] L.S. Furtado and S.R.B. Oliveira, A teaching proposal for the software measurement process using gamification: an experimental study, in: Proceedings of the IEEE Frontiers in Education Conference (FIE), IEEE, Uppsala, Sweden, 2020, pp. 1-8. https://doi.org/10.1109/FIE44824.2020.9274194.

[6] M.M. Alhammad, A.M. Moreno, Gamification in software engineering education: A systematic mapping, Journal of Systems and Software, 141, 2018, pp. 131-150. https://doi.org/10.1016/j.jss.2018.03.065.

[7] G. Rong, H. Zhang and D. Shao, Applying competitive bidding games in software process education, in: Proceedings of the 26th International Conference on Software Engineering Education and Training (CSEE&T), IEEE, San Francisco, CA, 2013, pp. 129-138. https://doi.org/10.1109/CSEET.2013.6595244.

[8] E. Navarro, SimSE: A Software Engineering Simulation Environment for Software Process Education, Ph.D. thesis, Donald Bren School of Information and Computer Sciences, University of California, Irvine, 2006.

[9] C. Sims, H. L. Johnson, and Agile Learning Labs, How to play the Team Estimation Game, 2012, URL: https://agilelearninglabs.com/2012/05/how-to-play-the-team-estimation-game/.

[10] B. Marin, M. Vera, and G. Giachetti, An adventure Serious Game for Teaching Effort Estimation in Software Engineering, in: Proceedings of the International Workshop on Software Measurement and International Conference on Software Process and Product Measurement (IWSM/MENSURA), CEUR Workshop Proceedings, Haarlem, The Netherlands, 2019, pp.71-86.

[11] E. Jääskä, and K. Aaltonen, Teachers' experiences of using game-based learning methods in project management higher education, Project Leadership and Society, vol. 3, 2022, pp. 1-12.

[12] S. Ouhbi and N. Pombo, Software engineering education: Challenges and perspectives, in: Proceedings of the IEEE Global Engineering Education Conference (EDUCON), IEEE, Porto, Portugal, 2020, pp. 202-209.

[13] T. Hacaloglu, B. Say, H. Unlu, N. K. Omural, O. Demirors, A Survey on COSMIC Students Estimation Challenge, in: Proceedings of the 31st International Workshop on Software Measurement and International Conference on Software Process and Product Measurement (IWSM/MENSURA), CEUR Workshop Proceedings, Izmir, Turkey, 2022.

[14] A. Abran, Software Project Estimation: The Fundamentals for Providing High Quality Information to Decision Makers, Wiley & IEEE-CS Press, Hoboken, New Jersey, 2015.

[15] COSMIC, The benefits of COSMIC software sizing, 2022, URL: https://cosmic-sizing.org/cosmic-sizing/intro/benefits-of-cosmic/.

[16] ISO/IEC 19761: Software engineering - COSMIC: a functional size measurement method, ISO, Geneva, (2011, reviewed and confirmed in 2019).

[17] C. Commeyne, A. Abran, and R. Djouab, Effort estimation with story points and COSMIC function points - an industry case study, Software Measurement News, vol. 21, no. 1, 2016.

[18] M. Salmanoglu, T. Hacaloglu, and O. Demirors, Effort estimation for agile software development: comparative case studies using COSMIC functional size measurement and story points, in: Proceedings of the 27th International Workshop on Software Measurement and International Conference on Software Process and Product Measurement (IWSM/MENSURA), ACM, Göteborg, Sweden, 2017, pp. 41-49.

[19] C. Symons, A. Abran, C. Ebert, and F. Vogelezang, Measurement of Software Size: Advances Made by the COSMIC Community, in: Proceedings of the 26th International Workshop on Software Measurement and International Conference on Software Process and Product Measurement (IWSM/MENSURA), IEEE, Berlin, Germany, 2016, pp. 75-86. doi: 10.1109/IWSM-Mensura.2016.021.