Nurturing Collaboration in an Undergraduate Computing Course with Robot-themed Team Training and Team Building

Michael James Scott
Games Academy
Falmouth University
Cornwall, UK
michael.scott@falmouth.ac.uk

Alcwyn Parker
Games Academy
Falmouth University
Cornwall, UK
alcywn.parker@falmouth.ac.uk

Brian McDonald
Games Academy
Falmouth University
Cornwall, UK
brian.mcdonald@falmouth.ac.uk

Gareth Lewis
Games Academy
Falmouth University
Cornwall, UK
gareth.lewis@falmouth.ac.uk

Edward J. Powley
Games Academy
Falmouth University
Cornwall, UK
edward.powley@falmouth.ac.uk

ABSTRACT

Group projects are a common feature of undergraduate degree programmes in computing. Early and sustained collaboration helps students to strive beyond introductory programming towards professional software development. However, during their first year of study, students can find teamwork challenging. To equip learners with the foundational knowledge, skills, and experience that they need to collaborate effectively so early in their studies, a 3-day Robot Olympics using *Lego Mindstorms EV3* robots can be deployed. The exercise draws upon Salas' big-five model of teamwork, making first-year students aware of coordinating mechanisms that aid in clarifying expectations and managing conflicts. These then act as lenses for reflection and feedback. Comparing a baseline cohort in 2015-16 to a cohort in 2016-17, after the introduction of the Robot Olympics, reveals a statistically significant reduction in team discord in an assessed collaborative programming project ($d = 0.76$). This suggests that the Robot Olympics made a positive contribution to the design of the first computing module. Notably, helping students to enact and reflect upon their group work and related employability skills.

CCS CONCEPTS

• **Software and its engineering** → Programming teams; • **Social and professional topics** → Software engineering education;

KEYWORDS

Collaboration, Group Projects, Teamwork, Programming, Robots

CEP '19, January 9, 2019, Durham, United Kingdom

ACM ISBN 978-1-4503-6631-1/19/01…$15.00
https://doi.org/10.1145/3294016.3294019

ACM Reference Format:
Michael James Scott, Alcwyn Parker, Brian McDonald, Gareth Lewis, and Edward J. Powley. 2019. Nurturing Collaboration in an Undergraduate Computing Course with Robot-themed Team Training and Team Building. In *Computing Education Practice (CEP '19), January 9, 2019, Durham, United Kingdom.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3294016.3294019

1 INTRODUCTION

Undergraduate students often experience discord when confronted with the complexities of collaboration, particularly when managing a group software development project for the first time. Obstacles include: coordinating times to meet; tracking task progress and blockers; communicating effectively; resolving conflicts; and collaborating effectively in the face of (sometimes, wildly) differing abilities [14, 16]. This not only influences satisfaction [10], but can also widen experience gaps as pressure to perform influences task allocation in ways that may deprive some group members from learning opportunities [3].

Despite these obstacles, group projects are a prominent feature of computing education. A 1997 survey of higher education institutions in the United Kingdom (UK) revealed that group projects were a key feature of provision across the sector [19]. Wiggberg also notes that "universities in the Western world largely organise computer science education in such a way that group work is an integral part of the students' education" [37, p. 21].

Such prominence is unsurprising. Employers cite that "soft skills" are critical [2, 26]. Talent supply is as an ongoing concern in the UK, with 25% of hubs describing hiring as a "major challenge" [33]. Perhaps driven by expectations that those entering the job market should be able to "hit the ground running" [4].

This is spurring efforts to bridge work and pedagogic practice to ensure that learners develop employability skills [27]. However, educators argue that students are unlikely to master teamwork through mere *ad-hoc* experience [21] and so structured exercises are necessary [35]. These include: team training, team building, and coaching. However, little work examines these exercises in the context of first-year computing. This paper aims to address this gap by evaluating a robot-themed team building and training exercise deployed early into an undergraduate computing course.

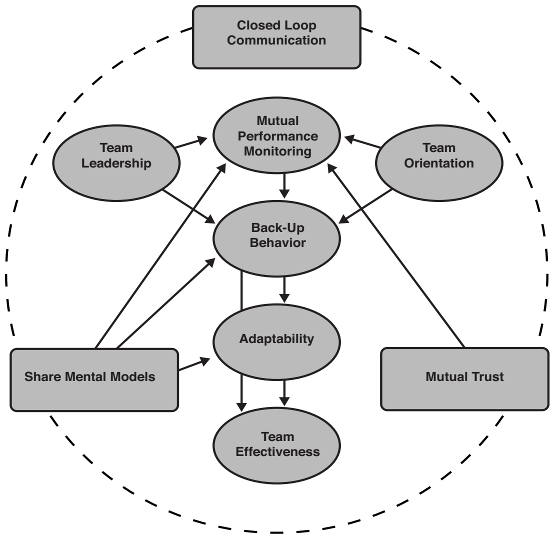

Figure 1: Five-Factor Model of Team Effectiveness with Coordinating Mechanisms (from [30])

2 BACKGROUND

A key motivation for embedding group projects into education is situated cognition. This is the notion that knowledge is bound to the activity and context in which it is learned [5, 6]. In other words, to learn practice that is meaningful to future employers, it should represent something that might be encountered during employment. Thus, projects aim to mirror the world of work—in some cases, doing so from the outset of a course [32].

The strategy also forms part of a wider trend in higher education (particularly, in software engineering) to develop and sustain learning communities [11]. Cooperative learning has long been considered effective [15]. It confers benefits such as knowledge sharing and mutual support, possessing synergies with other best practices in computing education (e.g., [25]).

Such approaches are also informed through insights into communities of practice. Notably, *cognitive apprenticeship* [7] and *legitimate peripheral participation* [18]. On the one hand, that modelling and articulating tacit knowledge and implicit processes is required to enact a complex skill and then improve it (i.e., observation, practice, and reflection). On the other, that newcomers first participate in low risk but nonetheless productive tasks, increasing in risk and value as they are inducted into the community and eventually work independently.

Towards this end, the design of scaffolding, drawing on the notion of the *zone of proximal development* [36], is pertinent. Following Hackman's [12] observation that discordant teams are unable to perform; in the absence of explicit guidance and supervision, students can worsen the conditions that make groups productive and ultimately successful. As such, structure and support when students collaborate for the first time is valuable. Typically, educators already ensure that tasks are designed carefully and groups are kept relatively small [20]. However, teamwork research offers a rich array of constructs [23] that educators should consider when designing scaffolds.

One such set of constructs, forming the basis for the practice described in this paper, is Salas' *Big Five Model of Teamwork* [30]. This model strives to condense the facets and relationships that influence team functioning into five key factors and three coordinating mechanisms, as shown in Figure 1. This model is sufficiently general to be extensible into many contexts, and handle many task-specific idiosyncrasies [9, 29].

For example, the model can be integrated into group projects in the first-year computing context through *team training* and *team building*. The former advances group members' knowledge, skills, and attitudes by "building cohesion, managing coordination, and enhancing communication" [p. 563] in a generalisable way [9]. The latter enhances social relations, helping a group to set goals, clarify roles, develop norms for managing interpersonal relations, and practice problem solving. Meta analyses show that both team training [28] and team building [17] enhance team outcomes.

3 CONTEXT

The practice described in this paper is integrated into *BSc(Hons) Computing for Games*, offered in the Games Academy at Falmouth University in the UK. By the end of their first year, students are expected to work effectively in multidisciplinary groups. Many projects are shared with art and design courses where students, typically in teams of 8-12 (of whom only 2-4 may study computing), collectively produce digital games. In preparation for multidisciplinary development practice, teamwork is heavily emphasised across the first study block of 15 weeks. Students are enrolled on an introductory programming module in which they solve problems through pair programming. They develop technical communication through exercises in their principles of computing module. They also embark upon a development principles module which covers agile project management, version control, and interpersonal skills.

The students enrolling on the course tend to arrive with a blend of academic and vocational qualifications, typically achieving 104-120 UCAS tariff points[1]. Approximately 43% of students report that they have no formal programming experience, with 50% having a pre-university qualification in computing. Their median age is 19 and there are few women (~3%). The intake has been stable, with no statistically significant differences between the 2015-16 and 2016-17 cohorts in terms of demographic variables such as age or gender, nor any measures of academic ability such as tariff points or prior programming experience.

In 2015-16, the course team observed many bad practices as small teams of computing students completed their first assessed collaborative programming project at the end of the first study block. A peer-review exercise confirmed high levels of discord within each team. This was concerning given the cohort would soon progress into a larger multidisciplinary project. In an attempt to overcome this challenge in 2016-17, the first week of the course was redesigned to incorporate team training and building through a 3-day Robot Olympics. To evaluate success, the module otherwise remained the same and the peer-review exercise was repeated to enable a comparison of team discord across the two cohorts.

[1]See https://www.ucas.com for a more detailed explanation of the tariff points used in the Universities and Colleges Admission System (UCAS).

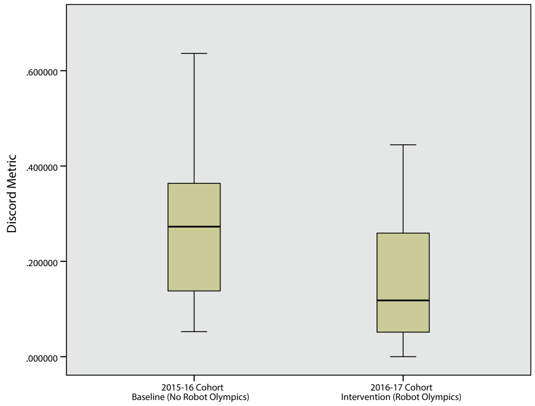

Figure 2: Box Plot Comparing Within-Group Discord Reported by Students in the Baseline (2015-16) and Intervention (2016-17) Cohort

4 PRACTICE

Inspired by similar olympiads reported in the literature (e.g., [31]), the distinction of the Robot Olympics presented here is its positioning: the very start of the course. It incorporates elements of team training and team building, orientating students using Salas' teamwork model as a framework to do so. It also offered an effective medium to teach agile methods [8]. During the Robot Olympics, students made use of the *LEGO EV3 Mindstorms* kit and completed challenges from the *Lego Mindstorms Space Challenge*[2].

The robot-theme was selected because robots are often associated with playful practical work rather than intense learning [1]. They present motivational affordances that capture curiosity and attention [24]. They have also been shown to be good ice-breakers in studies of transition to higher education, helping students build rapport [39]. Furthermore, a systematic review shows that 75% of studies published prior to 2012 found robots to be effective for teaching programming [22] and the visual block-based language used by *EV3* is accessible to even those students with no prior programming experience.

The *Lego Mindstorms Space Challenge* itself comprises a sequence of challenges in which group-made robots compete. These are presented to students as a set of user stories[3], progressing from initial setup (e.g., build the base robot), through to simple training missions (e.g. move in a straight line), through to space-themed activities that stretch each team's abilities (e.g., collect the asteroids). Each team is tasked with completing as many user stories as possible within the time constraint. Staff sign-off the setup and training user stories throughout the event, whilst the space-themed challenges are reviewed at an event on the final day.

There is an initial induction on: team-work strategies (e.g., communication strategies, conflict management, and mob programming); the robot's capabilities; and the user stories that form the space challenge. After which, students are sorted into teams of three. Staff and student mentors assume a facilitator role.

[2]https://education.lego.com/en-gb/support/mindstorms-ev3/space-challenge
[3]https://www.dropbox.com/s/he1g22zkslahpbn/Lego-Mindstorms-EV3.pdf?dl=1

Whenever a team completed a user story, or encountered a significant challenge, a retrospective was held privately with a supervising member of staff. Teams were observed throughout to highlight poor practice and/or disengagement. All of these opportunities strive to reinforce aspects of the five-factor model as described (on the next page) in Table 1.

5 ANALYSIS

Data was drawn from assessments conducted at the end of the first study block, including the peer-review exercises conducted by both cohorts. Each student rated their own engagement/contribution and that of their fellow team members (using a Likert-style response). The *SPA* statistic [38] was then calculated. Members of effective teams have *SPA* scores tending towards 1. Thus, differences from 1 can be used as a measure of discord:

$$\left| 1 - \sqrt{\frac{\text{Total ratings for individual team member}}{\text{Average of total ratings for all team members}}} \right| \quad (1)$$

An independent t-test examined the hypothesis that those students who participated in a Robot Olympics would report a lower (mean) level of discord in their first assessed collaborative programming project compared to those students who did not have the opportunity to do the Olympics. The results suggest a statistically significant difference ($p = .04$). The 2016-17 cohort experienced less discord in their subsequent group project ($\bar{x} = .164, \sigma = .134$) compared to the 2015-16 cohort who did not do the Olympics ($\bar{x} = .289, \sigma = .187$). The effect size is close to "large" ($d = 0.76$) and can be seen more clearly in Figure 2.

6 CONCLUDING REMARKS

Due to the observational and quasi-experimental nature of the way the practice is evaluated, confounding factors cannot be ruled out. Unobserved baseline differences could have influenced the levels of discord experienced by the teams. Nevertheless, the previous cohort experienced higher levels of discord in their group project in the absence of the Robot Olympics. The effect size is close-to-large, greater than Hattie's "hinge point" ($d > 0.4$) [13]. Given the theoretic grounding, no other substantial changes to curriculum or delivery, and no obvious ways in which the two cohorts differed, the difference could be attributed to the new practice.

Leveraging Salas' five-factor teamwork model (and agile concepts) visibly pushed students through the stages of team development in an intensive way, *storming* for less time [34]. Explicit reference to such models in discussions with teams seemed to help them to develop back-up behaviours and adapt when faced with challenges. The team building aspect aided the development of mutual trust, shared mental models, and closed-loop communication. Supervision meetings were initially frequent, but became less so as groups normed and performed.

Overall, the Robot Olympics helped students to overcome challenges associated with teamwork. This finding is useful to educators delivering computing courses with an emphasis on software development and which, to this end, position group projects early. However, the unique contribution of the robots remains unclear. As such, future work could explore this.

Table 1: Mapping of Facets of Salas' Model [30] to their Realization in the Robot Olympics

Facet	Implementation in the Robot Olympics
Team Leadership	Assign scrum master (managed retrospectives and stand-up meetings) and product owner (prioritised and coordinated user stories being worked on). Roles rotated each day.
Performance Monitoring	Review strengths and weaknesses of individual contributions and their relation to task strategy whenever an event is attempted. Retrospective held irrespective of success or failure.
Team Orientation	Shared team goal, set by the team and made explicit. Everyone acknowledging to work to this joint team goal, rather than their own goals.
Back-Up Behavior	Pair programming guidelines provided. Explicit team contract formed with if-then clauses (e.g., for disengagement). Teams identify their key skill areas and acknowledge to use this to inform pairings.
Adaptability	Pair up or mob to address identified weaknesses during tasks. Match-making of stronger and weaker students during tasks, and actively seeking of help when stuck.
Closed Loop Communication	Reminders to sit in a pod close to each other and to communicate often. Explicit structure of update reporting using periodic stand-ups.
Mutual Trust	Building of trust through first-day exercise. Reinforcement in each sprint retrospective to help students feel empowered to step up to tasks and deliver.
Shared Mental Models	Task board and flip-chart paper provided. Note keeping reinforced. Files kept on single shared computer per team.

REFERENCES

[1] Apiola, M., Lattu, M., and Pasanen, T. A. Creativity and intrinsic motivation in computer science education: Experimenting with robots. In *Proceedings of the Fifteenth Annual Conference on Innovation and Technology in Computer Science Education* (New York, NY, USA, 2010), ITiCSE '10, pp. 199–203.

[2] Archer, W., and Davison, J. Graduate employability: what do employers think and want?, 2008. https://aces.shu.ac.uk/employability/resources/0802gradempolyability.pdf.

[3] Barker, L. J. When do group projects widen the student experience gap? In *Proceedings of the 10th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education* (New York, NY, 2005), ITiCSE '05, ACM, pp. 276–280.

[4] Brindley, D. A big picture guy–interview with broadcom, inc., co-founder henry samueli. *Prism* (2001), 16–21.

[5] Brown, J. S., Collins, A., and Duguid, P. Situated cognition and the culture of learning. *Educational researcher 18*, 1 (1989), 32–42.

[6] Brown, J. S., and Duguid, P. Organizational learning and communities-of-practice: Toward a unified view of working, learning, and innovation. *Organization science 2*, 1 (1991), 40–57.

[7] Collins, A., Brown, J. S., and Newman, S. E. Cognitive apprenticeship: Teaching the craft of reading, writing and mathematics. *Thinking: The Journal of Philosophy for Children 8*, 1 (1988), 2–10.

[8] Devedzic, V., et al. Teaching agile software development: A case study. *IEEE Transactions on Education 54*, 2 (2011), 273–278.

[9] DiazGranados, D., Shuffler, M. L., Wingate, J. A., and Salas, E. Team development interventions. In *The Wiley Blackwell Handbook of the Psychology of Team Working and Collaborative Processes*, E. Salas, R. Rico, and J. Passmore, Eds. Wiley Online Library, Hoboken, New Jersey, 2017, ch. 24, pp. 555–586.

[10] Drury, H., Kay, J., and Losberg, W. Student satisfaction with groupwork in undergraduate computer science: Do things get better? In *Proceedings of the Fifth Australasian Conference on Computing Education - Volume 20* (Darlinghurst, Australia, Australia, 2003), ACE '03, Australian Computer Society, Inc., pp. 77–85.

[11] Froyd, J. E., and Ohland, M. W. Integrated engineering curricula. *Journal of Engineering Education 94*, 1 (2005), 147–164.

[12] Hackman, J. R. Why teams don't work. In *Theory and research on small groups*. Springer, 2002, pp. 245–267.

[13] Hattie, J. *Visible learning: A synthesis of over 800 meta-analyses relating to achievement.* routledge, 2008.

[14] Ikonen, J., Knutas, A., Wu, Y., and Agudo, I. Is the world ready or do we need more tools for programming related teamwork? In *Proceedings of the 15th Koli Calling Conference on Computing Education Research* (New York, NY, 2015), Koli Calling '15, ACM, pp. 33–39.

[15] Johnson, D. W., and Johnson, R. T. *Learning together and alone: Cooperative, competitive, and individualistic learning.* Prentice-Hall, Inc, 1987.

[16] Kinnunen, P., and Malmi, L. Do students work efficiently in a group? problem-based learning groups in basic programming course. In *Proceedings of the 4th Baltic Sea Conference of Computer Science Education, Koli Calling, Helsinki, Finland* (2004), pp. 57–66.

[17] Klein, C., DiazGranados, D., Salas, E., Le, H., Burke, C. S., Lyons, R., and Goodwin, G. F. Does team building work? *Small Group Research 40*, 2 (2009), 181–222.

[18] Lave, J., and Wenger, E. *Situated learning: Legitimate peripheral participation.* Cambridge university press, 1991.

[19] Lejk, M., Wyvill, M., and Farrow, S. Group learning and group assessment on undergraduate computing courses in higher education in the uk: Results of a survey. *Assessment & Evaluation in Higher Education 22*, 1 (1997), 81–91.

[20] LePine, J. A., Piccolo, R. F., Jackson, C. L., Mathieu, J. E., and Saul, J. R. A meta-analysis of teamwork processes: tests of a multidimensional model and relationships with team effectiveness criteria. *Personnel Psychology 61*, 2 (2008), 273–307.

[21] Lingard, R., and Barkataki, S. Teaching teamwork in engineering and computer science. In *41st IEEE Frontiers in Education Conference* (2011).

[22] Major, L., Kyriacou, T., and Brereton, O. P. Systematic literature review: teaching novices programming using robots. *IET software 6*, 6 (2012), 502–513.

[23] Mathieu, J. E., Hollenbeck, J. R., van Knippenberg, D., and Ilgen, D. R. A century of work teams in the journal of applied psychology. *Journal of Applied Psychology 102*, 3 (2017), 452.

[24] McGill, M. M. Learning to program with personal robots: Influences on student motivation. *Trans. Comput. Educ. 12*, 1 (Mar. 2012), 4:1–4:32.

[25] Porter, L., Guzdial, M., McDowell, C., and Simon, B. Success in introductory programming: What works? *Communications of the ACM 56*, 8 (2013), 34–36.

[26] Robles, M. M. Executive perceptions of the top 10 soft skills needed in today's workplace. *Business Communication Quarterly 75*, 4 (2012), 453–465.

[27] Ruff, S., and Carter, M. Communication learning outcomes from software engineering professionals: A basis for teaching communication in the engineering curriculum. In *39th IEEE Frontiers in Education Conference* (2009).

[28] Salas, E., DiazGranados, D., Klein, C., Burke, C. S., Stagl, K. C., Goodwin, G. F., and Halpin, S. M. Does team training improve team performance? a meta-analysis. *Human factors 50*, 6 (2008), 903–933.

[29] Salas, E., Shuffler, M. L., Thayer, A. L., Bedwell, W. L., and Lazzara, E. H. Understanding and improving teamwork in organizations: A scientifically based practical guide. *Human Resource Management 54*, 4 (2015), 599–622.

[30] Salas, E., Sims, D. E., and Burke, C. S. Is there a "big five" in teamwork? *Small group research 36*, 5 (2005), 555–599.

[31] Scott, M. J., Counsell, S., Lauria, S., Swift, S., Tucker, A., Shepperd, M., and Ghinea, G. Enhancing practice and achievement in introductory programming with a robot olympics. *IEEE Transactions on Education 58*, 4 (2015), 249–254.

[32] Shepperd, M. Group project work from the outset: An in-depth teaching experience report. In *Software Engineering Education and Training (CSEE&T), 2011 24th IEEE-CS Conference on* (2011), IEEE, pp. 361–370.

[33] Tech City UK. Technation report, 2017. https://technation.techcityuk.com/digital-skills-jobs/digital-skills-shortage/.

[34] Tuckman, B. W. Developmental sequence in small groups. *Psychological bulletin 63*, 6 (1965), 384.

[35] Vivian, R., Falkner, K., Falkner, N., and Tarmazdi, H. A method to analyze computer science students' teamwork in online collaborative learning environments. *Trans. Comput. Educ. 16*, 2 (Feb. 2016), 7:1–7:28.

[36] Vygotsky, L. S. *Mind in society: The development of higher psychological processes.* Harvard university press, 1980.

[37] Wiggberg, M. *Unwinding processes in computer science student projects.* PhD thesis, Uppsala universitet, 2008.

[38] Willey, K., and Gardner, A. Investigating the capacity of self and peer assessment activities to engage students and promote learning. *European Journal of Engineering Education 35*, 4 (2010), 429–443.

[39] Zarb, M., and Siegel, A. A. Facilitating first year student transitions into he computing using lego mindstorms. In *Proceedings of the 4th HEA Conference on Learning and Teaching in STEM Disciplines* (2016), HEA.