Fostering the Development of Computer Science Graduate Employability Through Agile Projects

Jordan Allison[a], Abu Alam[a], Luke Gassmann[b], Gareth Nelson[a], Kamal Zidan[a]

[a]University of Gloucestershire, Park Campus, Cheltenham, UK
[b]University of Bristol, Bristol, United Kingdom

ARTICLE HISTORY
Compiled April 2, 2024

ABSTRACT

This article presents the usage of Integrated Course Design (ICD) in the design and evaluation of applying agile methodologies within an undergraduate module of study to foster the development of computer science students employability skills. Undergraduate programs of computer science typically follow traditional educational methods which can lead to students unable to connect knowledge learned in class to actual situations and students are often assessed individually, whereas collaborative group projects are more akin to industry practice.

The teaching experience reported gives students the opportunity to relate concepts learnt in class to a practical group-based project. Students must meet the requirements of a 'client' who will provide feedback and additional challenges for students while following the Agile framework SCRUM. Positive student feedback and module grades 7.70% higher than the department average over a four year period indicates the teaching structure and assessment presented is an effective method to foster the development of technical and soft skills of undergraduate computer science students.

KEYWORDS
Computer Science Education; Agile Methods; SCRUM; Computing Pedagogy; Integrated Course Design; Employability

1. Introduction

The employability of computer science graduates is an ongoing concern, with a systematic literature review on employability for ICT graduates finding that while discipline-specific knowledge and skills are important, the non-technical (soft) skills such as communication, problem solving, project management, and business knowledge are agreed upon areas which require improvement (Aničić, Divjak, and Arbanas, 2017). These issues of computer science graduate employability are a particular concern within the UK, with the Higher Education Statistics Agency revealing how in 2017, computer science graduates were the most likely to be unemployed six months after graduation (HESA, 2017). Around this time, two majors reviews were released; the Wakeham Review of STEM Degree Provision and Graduate Employability (2016), and the Shadbolt Review of Computer Sciences Degree Accreditation and Graduate Employability

CONTACT Jordan Allison. Email: jrallison@hotmail.co.uk, ORCID ID: 0000-0001-8513-4646, LinkedIn: https://www.linkedin.com/in/jordanrallison/

(2016). Through a stakeholder survey, the Wakeham Review details how employers struggle to recruit graduates with adequate soft skills which include: project management, delivering presentations, entrepreneurial skills, report-writing, team-working, adaptability, and personal resilience (Wakeham, 2016). The Shadbolt Review paints a similar picture of the need for these types of soft skills, and suggests that there is more to be done to ensure that computer science students have the opportunity to develop such skills throughout their undergraduate degree program (Shadbolt, 2016). More recent reviews such as the Augar Review of Post-18 Education and Funding (2019), found that skill gaps have been highlighted in sectors such as Engineering and Computer Science, with more academic literature indicating that computer science graduates are not industry ready (Davenport, Crick, and Hourizi, 2020). Therefore, it is important to understand best practices to develop these soft skills in computer science degree programs (Joint Task Force on Computing Curricula: Association for Computing Machinery (ACM) and IEEE Computer Society, 2013). This is often the case in the design and teaching of software engineering principles through Agile methodologies. The Joint Task Force of the Association of Computing and Machinery and the Institute of Electrical and Electronics Engineers in their Curriculum Guidelines for Undergraduate Degree Programs in Computer Science, state that Agiles' "iteration-based, short-planning-cycle approach is a great fit for the reality of crowded undergraduate schedules and fast-paced courses" (2013), and that through such methods, students can develop a wide range of skills and apply software engineering principles more effectively (2013). With this in mind, this article will attempt to answer the following question:

> *How can Agile methodologies be utilised within higher education settings to foster the development of students technical and soft skills?*

To answer this question, we cover three criteria to analyse the application of simulated Agile-based business environments in an academic setting:

(1) Surrounding research for implementing Agile soft skill development in Higher Education.
(2) Implemented design for transferring and simulating modern business requirements to a Higher Education Computer Science course.
(3) Feedback covering the adaptability of course members, student soft skill development, and the quality of work achieved from the implemented course.

This article will be structured as follows. First, an overview of Agile will be provided, followed by examples from literature of how Agile has been utilised as a teaching approach. Next, through the methodology of Integrated Course Design by Fink (Fink, 2003), this article will explore the design and structure of how Agile was utilised as a module of study for second year computing undergraduate students within a UK higher education institution. Following this, an evaluation of the module will be provided with implications and recommendations for those who wish to adopt a similar approach.

2. Agile Methods

2.1. *Understanding Agile*

To answer our first criterion, we must evaluate modern transferable business principles. Agile methods are a collection of management and delivery frameworks, focused on

customer satisfaction and are delivered through innovative environments by early and continuous delivery of a viable product (Measey et al., 2015). Some examples of these frameworks include Extreme Programming (XP), Feature Driven Development (FDD), Scaled Agile Framework (SAFe), and Scrum, which will be the focus of this article.

To begin to understand a framework like Scrum, we must first look at its primary competitor which is a traditional framework; Waterfall. It is a linear, sequential approach that consists of several distinct phases, each of which must be completed before the next one can begin. The five phases of the waterfall methodology are: requirements gathering, design, implementation, testing, and maintenance. Each phase has a specific set of activities and deliverables, and the process flows from one phase to the next in a linear fashion. Waterfall methodology emphasizes a top-down approach, with the project manager or team leader setting priorities and making decisions at each stage of the project. Communication between team members is typically limited, and changes to the project scope or requirements are difficult to implement once the project is underway.

Scrum, proposed by Schwaber and Beedle in 2002 took a very different approach to software development as opposed to more traditional methodologies like Waterfall (Schwaber and Beedle, 2002). SCRUM is an agile project management methodology that emphasises iterative development, frequent testing, and continuous improvement. It is designed to help teams deliver high-quality products and services by breaking a project down into smaller, more manageable chunks called "sprints" Guenzel and Gillespie (2020). SCRUM is highly adaptable and flexible, making it well-suited for projects that are complex, uncertain, or rapidly changing projects. It emphasises collaboration, transparency, and regular communication between team members, and includes a set of roles, ceremonies, and artifacts to guide the team throughout the project life cycle. Some of these roles and ceremonies require more than technical development skills and require the ability to communicate effectively with both customers and developers, requiring a completely different skill-set. This is a very different approach vs a traditional project where a customer would outline their specifications in detail at the inception of the project and then wouldn't hear from a development team until the product was finished and ready to present (Mitchell and Seaman, 2009). User stories are short and simple descriptions of a feature from the user or customer's perspective, and these are created instead of requirements to focus on the product user (Schwaber and Beedle, 2002). These user stories are then ordered into a backlog based on factors like; business value, effort required and item importance. Scrum breaks software development projects into a series of time-boxed sprints where the development team take some of the user stories that are at the top of the backlog and begin working on implementing these (Schwaber and Beedle, 2002). The development team only work on the selected backlog items within their current sprint without adding or adjusting the sprint's workload. This allows for focused item delivery. The development team then present their work to the customer and stakeholders at the end of the sprint for feedback and comments. This feedback is then added and organised into the product backlog allow the customer to influence ongoing development in a structured way and the cycle would then begin again (see Figure 1).

2.2. *Agile Values and Mindset*

Agile methods require not only the technical skills in software development but also a number of soft skills such as mindset and certain personality traits to succeed. In their

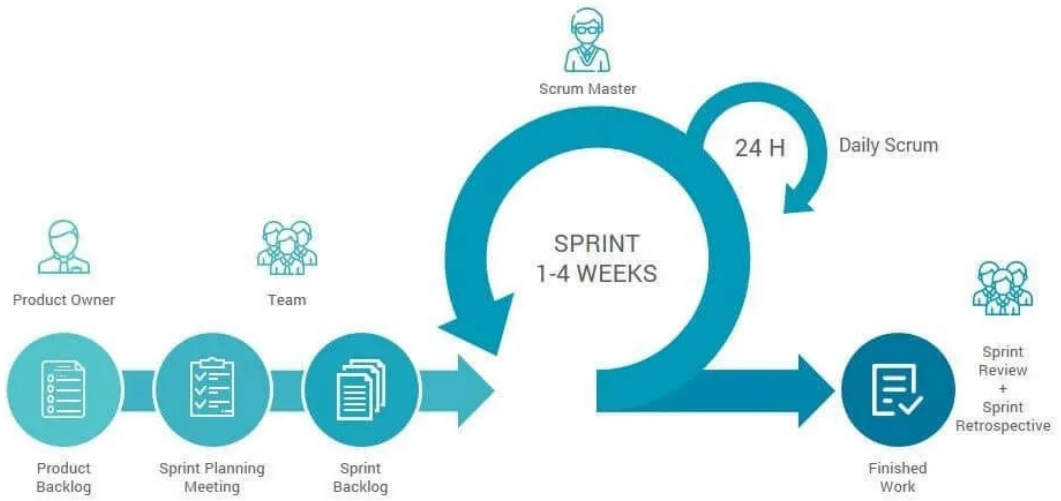

Figure 1. Scrum Framework. Adapted from Schwaber and Beedle (2002)

follow up paper published in 2014, Schwaber, Beedle and Sutherland felt that some of these values were so important that they wrote them directly into the agile manifesto (Beck et al., 2001). These values; focus, courage, openness, commitment and respect were required for teams to be able to work together and grow and deliver faster results and better products for customers/clients (Beck et al., 2001). This mindset allows the team to have a shared understanding of the project and a shared vision for the problems that the customer is attempting to solve (Babington, 2017). These values allow team members to challenge each other, build a shared understanding of strengths and weaknesses and allows the team to grow as a cohesive unit. This mindset is vastly different from a traditional methods mindset that is used in project methodologies like Waterfall where individuals would often work independently and then spend time integrating all of the individual work.

These team attributes demonstrated are a crucial component with feedback on a regular schedule from the customer/client (Cohn, 2010). This feedback shares the customers priorities and product vision. However, this adds an element of criticality to the work already produced (Cohn, 2005). These critiques, can improve the quality of the finished product and make it closer to the client's expectations saving the development team time should the requirements for the customer change. Change is embraced across all of the agile methods, especially SCRUM and allows the team time to develop their skills.

2.3. *Agile Projects as a Teaching Method*

Reflecting upon Agile Methodologies and their use in both industry and education, it's important when answering our first criterion to assess the resources and techniques used by other educational institutions. This is to review how Agile values manifest themselves in improving the standards of teaching, and the students' soft skills/career prospects, so that they can meet what would now be considered an industry standard (Baumann, 2020; Nyborg and Valente, 2021). This section will review past research into using and teaching Agile in higher education; focusing on communication, teach-

ing strategies, hybrid learning, and joining the gap between education and industry application.

Due to this incremental usage of Agile, computing educators are focusing on teaching students on how to adopt agile methods in practice. According to Neumann and Baumann (2021), the integration of eduScrum (an agile method in higher education) is a research gap that is under explored. Hence, they surveyed 21 students on a digital transformation course in Germany between 2019-2020 to identify how eduScrum can be used with real and practical problems in higher education, and how students value the work with their adapted version of eduScrum (Neumann and Baumann, 2021). Through using a five point likert scale from from 1 (completely applicable) to 5 (completely inapplicable), students from the Neumann and Baumann (2021) study gave the following scores:

- Overall, I rate the content of the course positive = 1.7
- The course content was conveyed clearly = 1.8
- My understanding of real world problems related to project management was promoted = 1.4
- My interest on the course objectives and content was aroused = 1.9
- The importance of the material for my professional activity became clear to me = 1.5
- I rate the didactic method (eduScrum) positively = 1.6

These are very positive results from Neumann and Baumann (2021) of using Agile in a higher education setting, but the survey feedback was conducted at the end of the course. Instead, it could have been performed in the first term to help fix issues prior to the start of the second term. For future studies, they recommend collecting discussion ideas with students (Neumann and Baumann, 2021).

The nature of a traditional classroom becomes disjointed when joined with the traditional principle of Agile Methodologies. The classroom traditionally involves an assembled hierarchy of teacher and learner, with the teacher seen as an authoritative figure to convey and clarify information, whilst Agile, at its rawest form, reflects a non-hierarchical structure that becomes most effective in environments with free-flowing control and discussion (Cristal, Wildt, and Prikladnicki, 2008; Stadler et al., 2019). In an attempt to mitigate this conflict of ideals, Nyborg et al (2021) proposes a hybrid-classroom approach where the teacher has the role of an unscripted expert in the field, whilst still being supported by traditional linear presentations. Meanwhile, the students are motivated by a more tailored educational experience, where any principles that have been misunderstood lead to easier tasks, whilst more confident students are given more challenging tasks. This is not dissimilar to the proposed hybrid learning by Sarikaya, Bagriyanik, and Gökalp (2020) which similarly implements a dynamic course based on student confidence and reflection, and mirrors the principles of Agile's sprint retrospectives by reviewing methods to improve understanding. Nyborg and Valente (2021) argue that this approach would be best supported by a tool that allows the lecturer to define and organise these tasks as slides and quizzes. This tool can then be used as data for student progression and more tailored education for each individual. This method of reflection also mirrors the feedback given during Agile Retrospectives. However, there are some constraints to this type of learning. Not only is there an issue with this system not fully reflecting self-learning, but instead assuming pre-made semi-static resources would improve learning more than face-to-face lectures. But also considering that Agile discourages over-planning, it is equally possible that a lot of

the pre-made resources will either not be used, or could have been replaced by a class discussion.

This restriction in class discussion and pre-made resource dependence is something that Sarikaya, Bagriyanik, and Gökalp (2020) also reflects on in their research. They propose that social and economic impacts affect the teaching within the classroom to a degree that traditional static teaching becomes less effective. In a similar attempt to further involve students in teaching and to provide a more tailored learning experience, Sarikaya et al (2020) proposes a different approach to Nyborg et al's (2021) individual feedback approach. This method has students taking an active role in the ongoing development and preparation for the module, which involves class discussions and negotiations, as well as, self-organised teams and resource management. The study allowed students to reflect on the course's positives and prepare for the next Sprint of learning to then present to the class to receive feedback and negotiate the course's future. This is not dissimilar from one of the activities used in Baumann's research (Baumann, 2020), as they implement a flipped classroom activity that has each group introduce the next module topic. This provides students with an opportunity to improve public speaking skills and become more active in their learning. Sarikaya et al (2020) class's continuous feedback and teamwork were maintained over the 15-week program with each sprint lasting 4-5 weeks. The resulting study found that both participation and enrolment in the course increased, as well as student motivation being linked to their perceived impacts within the classroom.

As shown by Sarikaya et al (2020), communication became a vital soft skill for the students to learn and was promoted through negotiation and visible changes in the course. Communication is further incentivised by Datta et al's (2020) research as they consider this to be essential for problem-solving and distinguishing between the developer and the intended product. With this in mind, Datta et al (2020) implemented 35% of the total module to rely on communication, with four main communication-related assignments. These included, developing a software development process for a hypothetical scenario, applying the agile manifesto to their hobbies, a lightning talk, and a retrospective reflection assignment (Datta and Mirza, 2020).

Agile is an active and dynamic framework, with communication and team negotiation being important aspects. It is therefore logical that interactive classroom activities provide a more engaging and active element to learning and team building. Briefly covered were activities implemented in Baumann's research (2020) like flipped classrooms, which provided students public speaking opportunities to introduce new topics to the class. Another activity suggested by Baumann's (2020) included Planning Poker, an Agile principle, where team members need to estimate the amount of effort an aspect of a project requires. The Scrum From Hell saw team members take on a positive and/or negative trait, and then role-play those attributes in a small team project, to learn negotiation skills with difficult clients and team members. To teach Agile's Minimal Viable Product ethos, Baumann's (2020) had students create a backpack list for a dystopian scenario and reduce this list to survive. This allowed students the opportunity to improvise and problem-solve to achieve the most with as little work as possible.

3. Materials and Methods: Creating an Agile Module of Study

Our second criterion for developing business simulated module design, includes deciding on learning activities that will be undertaken by the students to achieve the

learning outcomes (Young and Perović, 2016). Academic staff need to deal with a diverse student group, at the same time they want to develop and modernise their practice through designing and delivering successful courses and modules (Mcinnis, 2000). A comprehensive level of planning is required for teaching and learning activities to lead to a valuable learning experience for students (Fink, 2003). Donnelly and Fitzmaurice (2005) suggest how to develop the coherence of module design using learning outcomes, teaching methods, materials and activities, and assessment. They further explain that modules cannot be designed in an isolated way but need to fit within a course or program structure (Donnelly and Fitzmaurice, 2005). One way of doing this is by following a methodological framework of course design. As a basis for our second criterion, we review Finks twelve steps of integrated course design (Fink, 2003) as one framework for doing this.

Finks integrated course design (Fink, 2003) was used as it focuses on ensuring an integrated approach of learning goals, teaching and learning activities, and feedback and assessment. When created in isolation, these aspects may not link together effectively, and so documenting the steps used for integrated course design ensures this is done sufficiently. This was important in the context of the module as this integration links to constructive alignment (Biggs, 2003), which forms part of the universities assessment strategy of where the module is delivered.

Some authors have highlighted the importance of how courses are designed in higher education (Jahnke and Liebscher, 2020), and by presenting the steps of the course design process through integrated course design, this allows readers to comprehensively understand what was done so they are able to adapt the approach to their own practice. Furthermore, Finks integrated course design (Fink, 2003) has been used previously in other technology related courses for improving student learning and course delivery (Zhang et al., 2012), whilst some authors have concluded that compared to other course design frameworks, Finks incorporates many features of other approaches (Ramnath and Hoover, 2016). Hence, Finks integrated course design is used to outline the design and development of the Agile module.

3.1. *Step 1: Identify Situational Factors*

The first step of Finks integrated course design (Fink, 2003) involves outlining the course context. The agile module of study (titled 'Agile Methods'), is a second year module for students undertaking a degree in Computing, Business Computing or Computer Science. Taking place over the entire academic year in the academic year 2021-2022 (and likewise for the previous three academic years), this module was timetabled for a face-to-face three hour class once a week in a computing lab for twelve weeks each semester, with a cohort of 19 learners (2020-2021 n=19). Students would begin this module with the underpinning knowledge and skills gathered during their first year of their undergraduate study. This included topics such as computers and security, an introduction to programming fundamentals, an introduction to web development, and systems design and development.

As shown in Figure 2, students would partake in the Agile Methods module alongside two other compulsory modules per semester, and one optional module. For example, in semester one, students studied the compulsory modules of Agile Methods, Operating Systems, and IoT Development, and could choose between either Data Analytics or Network Design and Configuration. All of the four modules per semester had a singular piece of coursework each, and were due in the assessment periods of December-January,

and May-June. For each semester, the Agile Methods module was worth 25% of a students overall grade for the academic year.

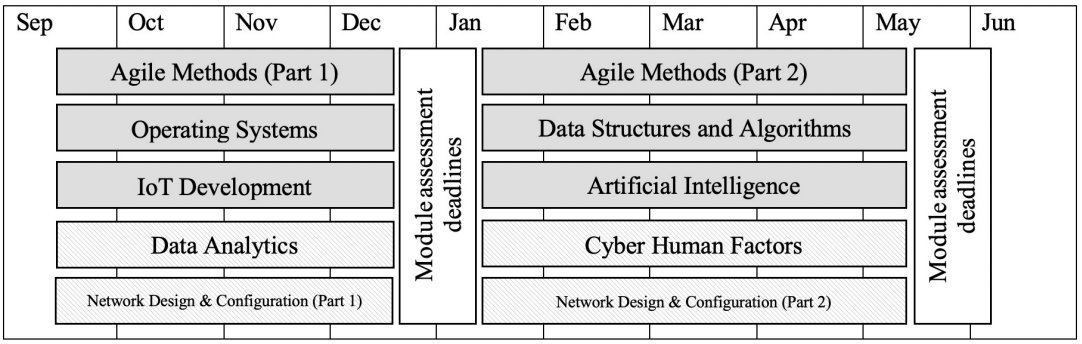

Figure 2. Student Module Timetable

The Agile Methods module was delivered by two module tutors, where one has experience as a software developer, a senior software architect, freelance software engineer, and project consultant. Meanwhile, the other module tutor has worked as a business IT consultant prior to university teaching, in addition to having extensive experience regarding Agile methodologies. Other lecturers also provide support to the module on an ad-hoc basis.

3.2. *Step 2: Identify Learning Goals*

The South-East England Consortium for Credit Accumulation and Transfer (SEEC) suggests some cognitive skills a level 5 student should have, where one is to critically evaluate concepts and evidence from a range of sources (Raiker, 2010) and so this is reflected in the learning outcomes:

- Critically evaluate alternative, traditional process-driven approaches with agile models and techniques within the context of appropriate software and business development environments
- Propose and defend innovative solutions to software development and associated business problems using appropriate agile software development strategies
- Discover the challenges to adopting agile strategies within database application development
- Learn about proven agile database techniques that enable fast, high-quality evolutional database application development

3.3. *Step 3: Formulate Feedback and Assessment Procedures*

While feedback is ongoing throughout the module, where all tutors offer formative feedback on assessments during timetabled sessions, and in sprint review meetings (later discussed), the assignment within the Agile Methods module requires a team (of 5 students) to develop a Database Application by following agile development principles, with Scrum as the adopted methodology. A key aspect of this assignment is the demonstration of good project management that facilitates agile development and team evolution. The team will increase their mark potential if it shows good project management by gaining equitable team-member contribution throughout the assign-

ment. The assignment assesses the team's ability to adopt agile development methods to produce a business system that recognises high-value business processes, a realistic appreciation of software development tools and their capabilities, and proficiency in turning business requirements into useful software applications. The assignment's scenario is realistically complex and challenging, with the summary of the scenario outlined below.

Assessment 2 Scenario: Emergency Water Bowser Scheduling, Deployment, Notification and Maintenance Information System

Assessment Brief: Your team is looking to earn its living by developing and selling to UK water companies an information system that will assist them in their deployment and management of water bowsers, which they need to deploy during emergencies and disruptions to water supplies. There is a trade show at which you want to demonstrate a working system in order to procure orders. The system you want to demonstrate at the trade show will be a fully functioning prototype for a fictitious generic water company. Key roles for the proposed system are:

(1) Assisting in planning the locations for bowsers.
(2) Recording information about the deployment of the bowsers.
(3) Tracking bowser maintenance requests and logging mechanical problems.
(4) Scheduling and monitoring bowser refilling.
(5) Informing consumers of bowser locations, as well as the status, refill schedule and perhaps other information that the water company might decide is appropriate.
(6) Enabling bowser problems to be reported and recorded, monitored and processed.

Although practical, the assignment requires full documentation and evidence of reflective review. The documentation must explain all that the team have done and confirm their technical competence and agile project-management ability. Each team member writes an individual statement explaining how they contributed to the team; what tasks they worked on; what they learned; and why they feel they contributed equitably to the team workload. The assessment is designed to help students develop both technical and analytical skills, and to communicate effectively through written reports and presentations, and to work both individually and as part of a team. All of these skills which were vitally highlighted in the Wakeham Review (2016) In summary, the assessment takes the form of case scenarios, evaluation of business requirements, identifying and analysing the Software Engineering solutions to a given problem and identifying measures required to ensure a viable environment.

3.4. *Step 4: Select Effective Teaching and Learning Activities*

Overall, the aim is for students to take charge of their own learning and make more informed choices about the options they take on their course. In the indicative syllabus, the most recent and adopted technologies/concepts in the subject area were added:

(1) Design of software
(2) Programming framework
(3) Database design concepts
(4) System analysis approaches such as defining user stories and condition of satisfaction (Schwaber and Beedle, 2002)
(5) Playing planning poker for task estimation process (Baumann, 2020)

(6) Using tools such as Jira for project management (Garcia et al., 2020)

Utilising these technologies/concepts ensured students can learn employability skills. The Higher Education Academy provides guidelines about how to develop employability skills (Grice and Gladwin, 2004). In general, subject related knowledge and understanding is developed through lectures, tutorials, computer-based and workshop laboratory sessions and guided independent study. The level of guidance and support varies as the student progresses through their course. Time pressure situations are experienced throughout this module involving team work. The emphasis is very much on the cumulative effects of gaining knowledge. Presentations are used extensively to gauge student learning. It should be noted that assessments are generally challenging in Computer Science and require substantial independent learning from students for them to achieve good quality grades (Dichev and Dicheva, 2013). However, students could gain feedback along the way on their assessment through sprint review meetings, as would happen in industry.

As a key part of the scrum framework, sprint review meetings play an important role (Schwaber and Beedle, 2002), and it was intended for learners to partake in sprint review meetings to help them utilise agile methodologies, develop their group project, practice presenting and the ability to answer questions, and to receive ongoing feedback. To incorporate this into the module, sprint review meetings would include students creating a presentation in their groups, and to provide a demo of their project to a 'client'. The client being an academic member of staff who would role play as a customer for the associated assessment 2 scenario, such as the CEO of a water supply company. After the presentation and demo, the 'client' would ask questions, suggest improvements, and provide general feedback to the students. This session would last approximately 90 minutes, and students would partake in two sprint review meetings.

3.5. *Step 5: Ensure Primary Components are Integrated*

The fifth step of Finks integrated course design (Fink, 2003) involves ensuring that the four previous steps are in alignment and appropriate to one another. This very much follows a constructive alignment approach which consists of ensuring that learning outcomes, teaching and assessment are aligned to each other (Biggs and Tang, 2011). Given the explict focus on the course design being integrated, the authors consider that situational factors, learning goals, teaching and learning activities, and feedback and assessment methods are in alignment.

3.6. *Step 6: Create Thematic Structure of Course*

This involves dividing the module into segments that each focus on key concepts or course topics (Fink, 2003). Fink suggests having four to seven segments, and for this module, the course was divided into roughly six segments: Theory - Introduction to agile; Theory - Overview of scrum; Theory - Sprint planning and reviews; Practical - Creating project plan; Practical - Creating project; Practical - Sprint reviews and system testing.

3.7. *Step 7: Select/Create Instructional Strategy*

An instructional strategy dictates how learning activities should be arranged so student knowledge accumulates as they go through the course (Fink, 2003). The instructional strategy is therefore aligned with that of the thematic structure of the course, with practical elements coming after the theoretical.

3.8. *Step 8: Create Overall Scheme of Learning Activities*

The scheme of learning activities to be discussed will focus on the second half of the module only (semester 2) where the focus is on the practical implementation of Agile concepts, whereas the first semester focused on theoretical content only. In semester two, the first teaching session starts with setting up the group, discussing the concepts of Agile Principles, team building, project management and identifying user requirements and write the requirements in user story format. After the lecture, each team identify the requirements for the application and then present to the class. The lecturer and teams participate in the discussion and make sure the understanding of the requirements are clear. The teams need to decide which business functions they feel should be given build priority based on business value.

In the second session, teams get familiar with a project management tool, JIRA which is widely used in industry (Garcia et al., 2020). Then the teams apply task estimation best practices (playing planning poker) for implementing the requirements and assign tasks to team members. Teams demonstrate the task estimation process to the whole class using an online tool and answer any questions from the lecturers or other teams. All these activities are logged within Jira. Teams also create a Product Vision box and complete a team vision form, both are useful activities for team building and motivation to build a good product (Benassi, Amaral, and Junior, 2011). All the activities within this module are time-boxed which means all tasks should be completed within an agreed time as it is one of the principles of Agile Methods (Schwaber and Beedle, 2002). If a deadline is not met, the adjustment in the plan should be justified.

In the third session, the principles of User Interface design are explored. Then, the students translate the requirements to a wireframe deign and present to the whole class. Adjustments on the wireframes are made based on the feedback given by the lecturers and other teams. The teams also exercise competitor benchmarking activities to identify what other people are doing and how well they are doing. This is also presented to the class.

In the fourth session Database Design principles are explored and database is designed for implementing the requirements of first phase (sprint 1). Again, the teams present their designs to the class and everyone including lecturers ensure the database design is correct and ready for integrating with other layers (front-end and business) as they are developed. Then the fifth and sixth session involves introducing design patterns such as MVC to design and implement a robust, scalable, and maintainable software product. The latest and widely-used frameworks such as Laravel used by industry are introduced (Soegoto, 2018).

In the seventh and eighth sessions students present their implementation of the first phase in the first sprint review meetings. Here, students would need to adjust their project plan based on the feedback given by the reviewers. In the ninth and tenth sessions, real-life testing approaches are demonstrated and industry standard testing tools such as selenium automated testing are taught (Razak and Fahrurazi, 2011). Finally, the eleventh and twelve sessions are the presentation of their phase 2

implementation (sprint review meeting 2).

3.9. *Step 9: Develop Grading System*

Defining a suitable formative and summative assessment strategy requires a deep approach to learning regardless of subject matter or institutional setting (Donnelly and Fitzmaurice, 2005). Meanwhile, a key component of following a constructive alignment approach is ensuring the grading of assessments is applicable and in alignment with learning outcomes and teaching activities (Biggs and Tang, 2011).

The Assessment of Agile Methods module is both formative and summative. Formative assessment keeps track of how the learner is progressing. During the sprint review meetings academics provide feedback to students which improves both the learning and the teaching practice. At the end of a module, summative assessment is applied to grade students' work. To make a learner-centred assessment, a portfolio type assessment for the final assignment is selected with two sprint review assignments during the semester. The final portfolio includes - individual statement for each team member explaining how they contributed to the team and evidence that the team has understood and adopted Scrum and agile, iterative development. Evidence should consider how the team prioritised the high value business functionality, analysed requirements, workload estimation, burndown rates, notes from the sprint review and retrospective meetings, and details of technical implementation. The assessment criteria is given in the Table 1.

3.10. *Step 10: De-Bug Possible Problems*

One disadvantage to Agile based and general group work is the social impacts that impact final grading. Agile groups are assembled using a self-organised team philosophy. This provides businesses with functioning groups that are built by developers to best mould the problem. However, in a university environment, students do not have the same incentives to work when placed in a group. Group grades are based on a collective effort, with other subjects competing for time. Thus, many groups have a *prisoners dilemma*, where individuals do not contribute, apply effort elsewhere, and receive the same grade as the rest of their group (Chang and Brickman, 2018). Based on the current approach, once a group is assembled, the students cannot leave or remove group members during the assignment's development. One solution to partially mitigate this problem, is to have students comment on how they have contributed to the assignment within the assignment report.

3.11. *Step 11: Write Course Syllabus*

Once the course was designed, and the timetable set, this information was conveyed to students via their virtual learning environment, and lecture sessions via their student calendar. Other details such as the module learning outcomes, assessment, structure and general module information was conveyed to students in the first face-to-face session with students.

Table 1. Marking criteria for Agile Methods module.

Grade	Content
To achieve <30	Some requirements met, but very limited and not recoverable. Copyright violation.
To achieve <40	- Poorly structured report or most parts missing or no personal statement or no evidence of using Scrum and iterative, incremental development - Deliverables partially complete, e.g. inadequate system or poorly designed database.
To achieve 40+	- Report shows limited understanding of Scrum and iterative, incremental development - Personal statement shows how you collaborated equitably throughout the project across a range of technical, decision making and administrative tasks - System with limited functionalities - Correct or nearly correct database model - Implementation of user authentication - Implementation of one data input form and one report - Limited documentation in Code
To achieve 50+	- Report shows limited understanding of Scrum and iterative, incremental development - Personal statement shows how you collaborated equitably throughout the project across a range of technical, decision making and administrative tasks - System with limited functionalities - Correct or nearly correct database model - Implementation of user authentication - Implementation of one data input form and one report - Limited documentation in Code
To achieve 60+	As 50+ with - Personal contribution shows how you learned from mistakes and became a valuable asset to future development projects - Your application must be intuitive to use, offer business value, and include features such as: - Different types of Forms and Report pages - Different types of page input items such as Check Boxes, Radio Buttons, Dropdown List etc - Navigation - Breadcrumbs, Tabs, Navigation Bars etc - Helpful messages for the users
To achieve 70+	As 60+ with - a very good application of Scrum and agile project-management throughout the project; good team evolution and effective demonstration of methods to ensure team cohesion and focus; extended consideration of business functionality, business processes and system design; - Excellent report; all sections written to a high standard and very clearly structured. - comprehensive and relevant testing;

3.12. *Step 12: Plan Evaluation of Course and Teaching*

To echo an Agile Retrospective, and to improve the flow of information and feedback between module leaders and students, it was planned for feedback to be gathered from students to review how the course had challenged them, as well as how the students had found the implementation of resources and whether the module had contributed to their understanding of Agile Methods. In developing questions to ask, we based our questions around a principle shown in research by Neumann and Baumann (2021), promoting the idea of not only how the course's resources impacted the students, but predominantly focusing on the class delivery and the support this offered students. As a result, students were asked questions which could be broken down into the following categories: student and course information, module expectations, employability, challenging factors, positive and negative contributions, communication, module delivery. These questions were asked to students who are/have finished the course for their feedback, with the future intention to distribute a similar document to engage in students' expectations of the course before enrollment. Finally, this research did not require formal approval by the University's Research Ethics Committee due to the

Table 2. Agile Module and Department Assessment Results 2018-2022

Year	No. Students	Agile Mean	Department Mean	Difference
2018/2019 (F2F)	26 (22M/4F)	61.39	53.12	+8.27
2019/2020 (F2F)	23 (20M/3F)	60.04	51.99	+8.05
2020/2021 (Online)	19 (17M/2F)	55.18	51.66	+3.52
2021/2022 (F2F)	19 (17M/2F)	60.8	49.85	+10.95
4 year average		59.35	51.66	+7.70

research adhering to the ethical consequences of researching with human subjects, and by following the home institutions Research Ethics guidelines.

4. Results and Discussion

In answering this paper's research question, our third criterion overviews the resulting adaptability of course staff and students and the quality of learning provided from a simulated business environment. We assess this criterion using four phases of feedback: the *Sprint Reviewer Evaluation* provides insight into the adapted module teaching, the *Module Tutor Evaluation* reviews the assessment results and transferable module design, the *Exempler Student Work* oversees the assessment products developed by the students, and the *Student Feedback* answers how students found adapting to the module. Whilst global social-economic impacts prevent us from making an explicit comparison of student marking data across the module's previous years (due to changes from in-person to hybrid teaching), the results collected and reviewed in this section do show an improvement in student results for the Agile module within the duration of this study in comparison to the overall department assessment results.

As shown in Table 2, over a four year period of delivering the Agile Module in the way described, assessment results have consistently been higher than the average of all modules taught within the department. For the four academic years 2018/2019 to 2021/2022, the average assessments results for students on the Agile module was 59.35%, which is 7.70% higher than the department average of 51.66% over the same four year period.

In the year 2020/2021 the module average result was approximately 5% lower than the other three years of delivery. During this year the mode of delivery changed from face-to-face (F2F), and was instead delivered online via Microsoft Teams due to Covid-19. During this year, students were given the option of participating in the assignment in groups, or to attempt the assignment individually. Ten students formed three groups, and the remaining 9 students attempted the assignment individually, including solely taking part in the sprint review meetings. Although it is not certain, it is likely that the delivery of the module being online with less student interaction and teamwork led to less favourable module results during that academic year. However, general department results did not suffer so severely. Therefore, based on the evidence in this module, remote learning hindered student performance for modules with a heavy emphasis on teamwork.

4.1. *Sprint Reviewer Evaluation*

Sprint reviewers were asked to feedback on the process of being a sprint reviewer and how effective they think the process is to help aid student learning, and develop em-

ployability skills. One of the sprint reviewers commented *'I would like to express that I feel this type of assessment is excellent for students and is something that should be commended and used more frequently'*. Sprint reviewers discussed how it was clear how students improved their product from the first sprint review to the next, and indicated that the sprint reviews offer students the chance to develop skills such as presentation skills, teamwork, and responding to stakeholder requests. The sprint reviewers indicated how these skills are not usually developed in the modules of study which they teach and are seldom developed in computer science courses but are crucial for students to demonstrate in the workplace after university studies. In addition to this, Sprint reviewers were tasked with deliberately challenging the groups to make some changes to the work they have produced to further simulate how clients/customers might act upon presentation of the work the development team have produced and the students reaction to these changing requirements was also interesting to see. However, the sprint reviewers raised some concerns in that some students did not fully engage in the presentation process, while if the module had a much larger cohort of students, conducting many sprint reviews would become challenging. However, overall, the sprint reviewers discussed how they found the experience useful and beneficial themselves to gauge student learning and progress, in addition to being useful for aiding student development.

4.2. *Module Tutor Evaluation*

The assessment results were above the university average, and all students submitted an assessment, with one student failing the assessment (achieved less than 40%) due to handing in work that was incomplete. The Moodle feedback was also largely positive. The majority of the students stated that they enjoyed the group format of the assessment and emphasised that the use of Scrum to build an app was useful for them, allowing them to put the skills learnt from Semester 1 into practice. Students also found the assessment methods useful preparation for their job/placement applications and enjoyed the increased emphasis on technical skills too.

At the end-of-the semester, student works were evaluated to verify whether student work achieved the criteria and the learning outcomes were met. There was a general agreement that good marks in the assessment depended upon gaining a comprehensive understanding of Scrum/Agile by implementing the concepts practically and getting feedback about their progress with the assessment.

It is believed that Scrum/Agile approach can be useful for students for group work within other modules too and to deliver good assignments and to develop soft skills. Once students understand the Agile concepts from this Agile Methods module, they should be able to apply the knowledge to any other module with a group assignment by managing time effectively using a time-boxed approach and by communicating with team members in a timely-manner. More research is needed to prove this hypothesis. Feedback was collected from placement and graduate students, and there is a great deal of appreciation that the module helped students to work effectively in real-life scenarios when working in an Agile environment. Integrated course design was a key aspect in helping develop the module and by getting feedback from students and industry, and by measuring economic impacts, further improvements can be made.

4.3. *Exemplar Student Work*

This paper aims to elicit how Agile methodologies can be utilised within higher education settings to foster the development of students technical and soft skills. Therefore, we present an exemplar piece of work (see Figures 3 and 4) indicating a finished student product that resulted from undertaking the Agile module of study. It is difficult to understand the volume, quality and complexity of the whole implemented site, the professionalism and Agile practices of the group by looking into just two screenshots. Therefore, to give a better idea of the product implemented, the products' main features are outlined below:

- A designated public section that includes current effects in the area through a Twitter application programming interface (API), nearby bowsers and status with the ability to report a problem, and a FAQ page.
- A council and maintenance section containing a maintenance platform with access to the tasks page, a login page, bowser information, statistics, live feed and management, a report page, task creation, and a discussions page.
- An administrator section containing statistical reports for making business decisions, staff information management, constituencies management and a discussion board.

The module leader and sprint reviewers were well pleased with the work and trust that the quality of work is very close to real-life applications. Below is the summary of the feedback given to the group for their assignment (where they were awarded a mark of 85) and shows how the group adopted Agile throughout the project:

- The developed system is very good with functionality for both business processes and system management. The graphic user interface are designed very well and very usable. Many business processes have been very well covered across both the Water Company and the public interface, including operational and business monitoring provision.
- Good technical implementation of the following:
  - Usable User Interface
  - Security features such as password hashing, enforced password policy, input validations, access control and secured architecture
  - Extensive use of frameworks (such as Laravel, Google Map and Twitter API
  - Underlying data model and entity relationship
  - Scalable architecture
  - Automated Testing
  - Responsive and mobile friendly Application
- User Manual provides a useful step-by-step guide. It is written from the user's perspective and uses appropriate style.
- Agile Scrum was adopted very well. Very Good application of Scrum processes. User stories were clearly identified, evidence of sprint review meetings and retrospective meetings were provided. Burn down chart were produced and included in the report.
- Focused on tools and resources that benefited the team best.
- The group worked together very well and extensive use of the project management tool (JIRA) were demonstrated.
- Personal Statement demonstrates contribution to the project and awareness of

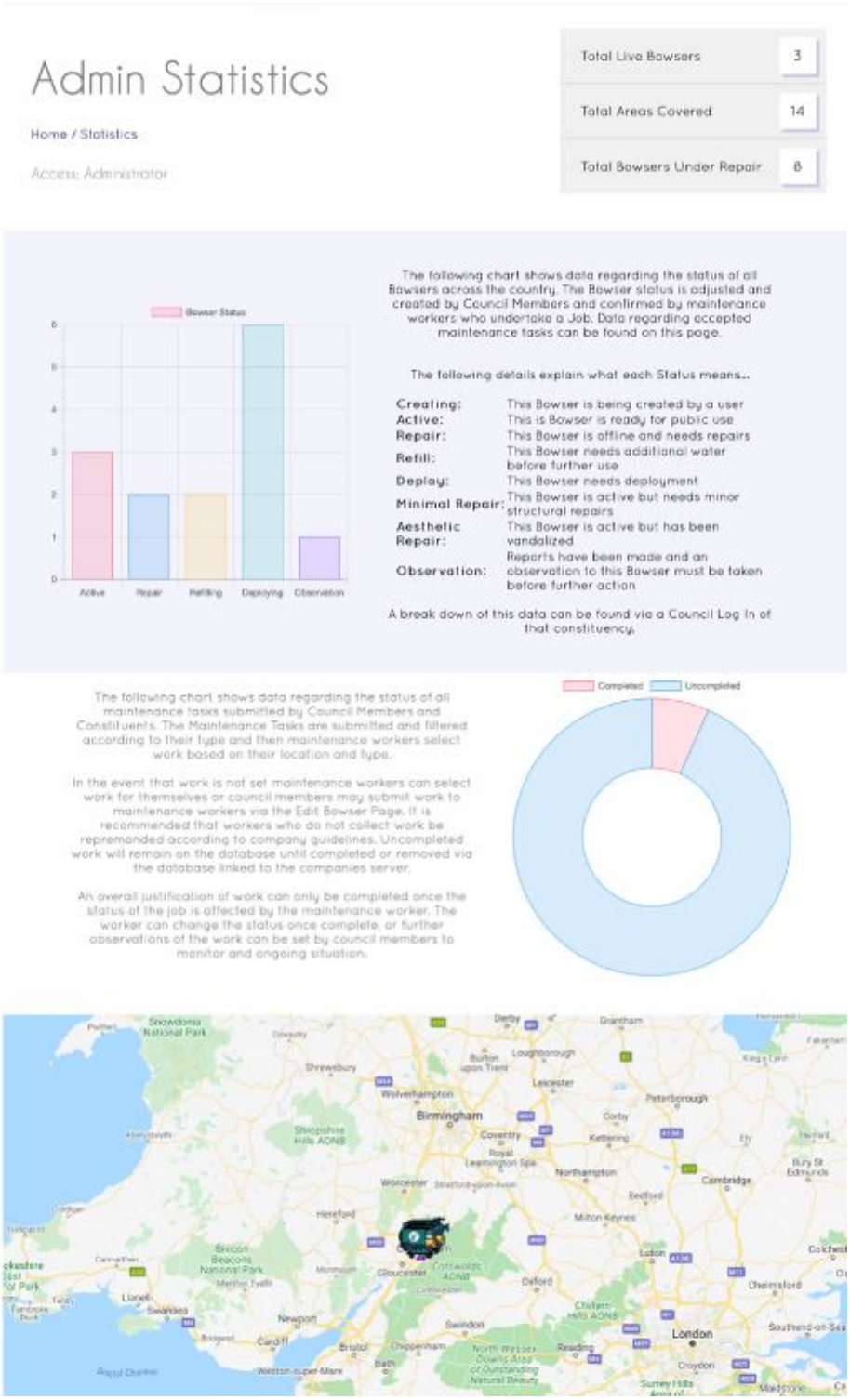

Figure 3. Example student work: visual representation for the admin to manage the employees

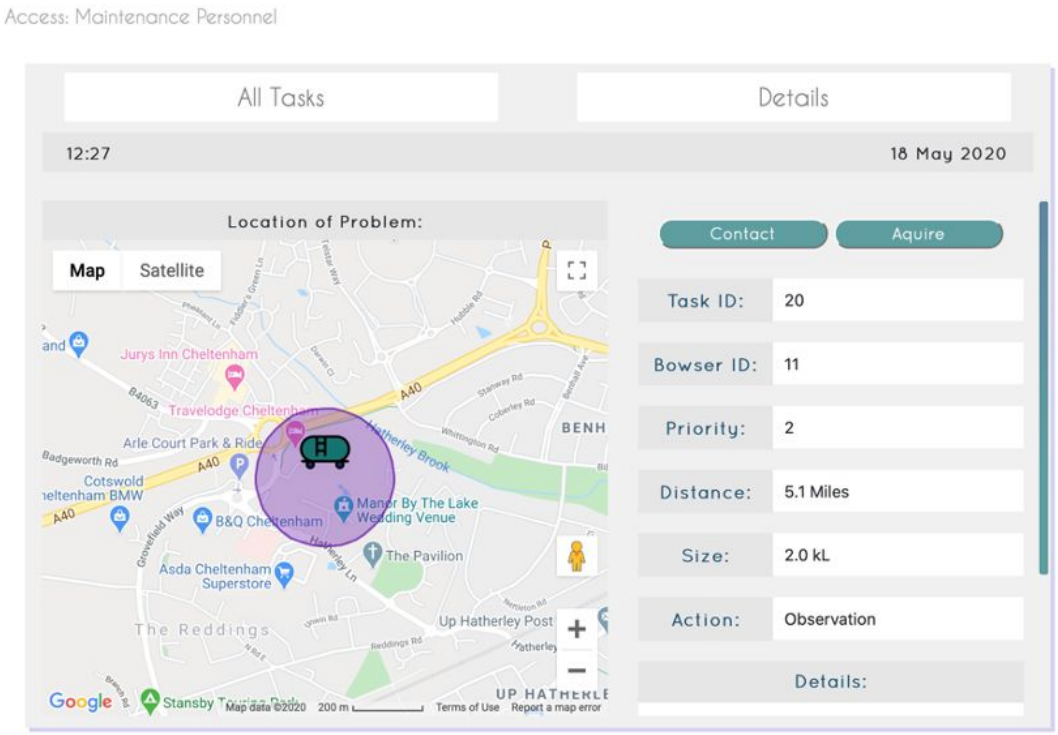

Figure 4. Example student work: A feature implemented for maintenance worker to view the location and other details of a problem

the development processes.

Without following any agile processes, it is likely that the developed system would not be as effective. During two sprint review meetings, the group not only had the opportunity to develop and receive feedback on their presentation skills, but they received 'client' feedback about the features implemented, the user interface, security and access control from a business perspective from the sprint reviewers at an early stage. Without this kind of practice, it is common that a software engineering team ends up with implementation of a Gold-Plated application (i.e. the engineers think they are implementing something useful but this is not true for the business). Also, early feedback helps student to meet their assignment deadline while implementing a good product.

By following Agile methodologies throughout this module of study, the module design and this exemplar work together demonstrate how students can effectively develop both technical and soft skills that are crucial for a career as a software engineer or in other related fields.

4.4. *Student Feedback*

Following the second sprint review meetings, each group was asked questions about how they have found the module, including how they found the usefulness of sprint review meetings in developing soft skills, completing their assignment on time, delivering a good product, working as a team, and if they think the method can be used and useful for other modules too.

One group of students discussed how they worked well together on the group assignment, and found tools such as Jira very beneficial. The group explained how they met once or twice a week via Microsoft teams for approximately 1-2 hours to discuss progress. One student commented that *'No team leader helped facilitate an open, honest discussion'*, while another explained that the *'Module pushes you to be more open and more sociable, and be more independent'*. These are clear benefits from the module delivery method, with this group also explaining how they feel they can use the skills from the module (i.e. Scrum) for employment and other modules. In fact, one student discussed how they have secured a work placement and they explained how they felt that discussing and working on this Agile project helped them secure that employment.

Another group of students revealed some similarities to the first group. They explained that they worked well as a team, where they have a group chat in place to *'bounce ideas about'*. They explained how they also had regular Microsoft Teams meetings so work could be done seamlessly and kept on track. However, this group did have one group member who was not engaging in the process, with the students commenting how it is stressful not being able to fully trust group members and that they *'Don't want to risk [their] grade on those who do not attend'*. Hence, they were somewhat sceptical about having modules adopting other group-work assignments. However, the students explained how the module has developed their soft skills, and in some cases helped them *'re-warm'* to some technical skills. One student commented how they have developed their confidence as the module has progressed and stated: *'I have broadened [my] skill set doing this module'*. The group also found the sprint reviews useful for gaining ongoing feedback, and they felt that they could apply agile methodologies to other modules, even on solo projects. Most importantly, the students explained how they have *'developed a more proactive approach to working'*, which they can utilise throughout the remainder of their course and in future employment.

Feedback from the student's after completing the module as a retrospective for future improvement. When asked about future employablity and soft skills gained during the assignments, one student argued that the module provided an edge to future work prospects. This was reinforced by two other students who said that they and their team gained cross-disciplinary skills like project management and project development. The structure of the client meetings were also discussed and were seen to have advantages over other module assignments due to the marking being spread across multiple meetings giving each team time to develop their skills over the year. One student discussed the advantages of meeting with the client as ensuring that the entire team were working consistently on the project. They did, however, point out that this became harder to demonstrate during back-end development due to the lack of new visible features. A student in the same group also agreed saying that client and weekly meetings provided additional clarity to the assignment/project criteria that otherwise would not have been available. When asked about team building the students said that by insisting on in-person discussions team-member relationships were reinforced and there was less miscommunication in comparison to other forms of discussion (text, social media). They also said that that in-person meetings improved team-building by providing skills in responding to feedback without external help. Furthermore, one of the students further concluded that the modules Agile structure would provide improvements to other modules *'as it can provide collaboration and assistance between individuals'*, however, another student pointed out the limitation of the assignment structure based on the modules output, stating that *'in modules where there is limited user interface... this would not be suitable'*.

5. Conclusion, Limitations and Recommendations

To answer whether Agile methodologies can be utilised in Higher Education to foster students' technical and soft skills, we developed a framework to provide students with a simulated business environment that has the incentive for team productivity and collaboration. The proposed framework incorporates Agile using strategies like Planning Poker (Baumann, 2020), group presentations and the prevention of over-planning (Nyborg and Valente, 2021), and focusing on the development of soft-skills like team communication (Datta and Mirza, 2020). Via student feedback the framework has shown success regarding team building and individual confidence which have provided a firm foundation for future careers, as well as being coupled with a better knowledge of implementing Agile strategies outside of the learning environment.

In relation to our structured criteria, this paper contributes to the body of knowledge regarding the use of Finks' Integrated Course Design by demonstrating how it can successfully be used for the development of teaching and learning activities within in a higher education setting where the focus is on software development using Agile methodologies. It is recommended that other practitioners consider how this course design method can be used in modules with similar content areas and learning outcomes. Furthermore, future research could aim to replicate the delivery of the Agile Methods module in other university settings for a comparative analysis. University courses are dynamic and designing courses needs an integrated approach by collecting feedback from practitioners, learners and industry.

A key limitation of this article is that students were not assessed on their technical skills or self-efficacy of soft skills before and after the module delivery. Therefore, it cannot be quantified how much the module design and assessment led to an improvement in these skills. Although the qualitative student feedback and subjective interpretation of student development was identified by practitioners involved in the module, future research should ensure that students are assessed prior to module delivery so that the effectiveness of approaches can more accurately be identified.

Reflecting on our third criterion, the experience and approach outlined here is not a one-time process, as the module should continue to develop based on feedback and experience of what has worked well (or not). Therefore, the experience documented in this article will inform the future design and delivery of the module for the academic year 2022-2023 and beyond. However, in providing a possible solution to our research question, practitioners can use the approach and assessment provided in this article to help inform their own course designs in teaching Agile methodologies and the development of student soft skills.

Disclosure statement

The authors report there are no competing interests to declare.

References

Aničić, Katarina Pazur, Blazenka Divjak, and Krunoslav Arbanas. 2017. "Preparing ICT Graduates for Real-World Challenges: Results of a Meta-Analysis." *IEEE Transactions on Education* 60 (3): 191–197.

Augar, Philip, Ivor Crewe, Jacqueline de Rojas, Edward Peck, Beverley Robinson, and Alison Wolf. 2019. *Review of Post-18 Education and Funding*. London: Department for Education.

Babington, G. 2017. "Being Agile: An evaluation of critical success factors in transitioning to Agile methods and the importance of the Scrum master in facilitating the Agile Mindset." .

Baumann, Annette. 2020. "Teaching software engineering methods with agile games." *IEEE Global Engineering Education Conference, EDUCON* 2020-April: 1647–1650.

Beck, Kent, Mike Beedle, Arie Van Bennekum, Alistair Cockburn, Ward Cunningham, Martin Fowler, James Grenning, et al. 2001. "Manifesto for agile software development." .

Benassi, João Luís Guilherme, Daniel Capaldo Amaral, and Lucelindo Dias Ferreira Junior. 2011. "Product vision management: concept and models evaluation." *Product: Management and Development* 9 (2): 163–172.

Biggs, John. 2003. "Aligning teaching for constructing learning." *Higher Education Academy* 1 (4).

Biggs, John, and Catherine Tang. 2011. *Teaching For Quality Learning At University*. 4th ed., Vol. 2011. Maidenhead: Open University Press.

Chang, Yunjeong, and Peggy Brickman. 2018. "When group work doesn't work: Insights from students." *CBE—Life Sciences Education* 17 (3): ar52.

Cohn, Mike. 2005. *Agile estimating and planning*. Pearson Education.

Cohn, Mike. 2010. *Succeeding with agile: software development using Scrum*. Pearson Education.

Cristal, Mauricio, Daniel Wildt, and Rafael Prikladnicki. 2008. "Usage of SCRUM practices within a global company." *Proceedings - 2008 3rd IEEE International Conference Global Software Engineering, ICGSE 2008* 222–226.

Datta, Soma, and Mahrukh Sameen Mirza. 2020. "Focusing on Both Teaching Agile Software Development and Communication." *2020 IEEE 32nd Conference on Software Engineering Education and Training, CSEE and T 2020* 279–280.

Davenport, James H., Tom Crick, and Rachid Hourizi. 2020. "The Institute of Coding: A University-Industry Collaboration to Address the UK's Digital Skills Crisis." In *2020 IEEE Global Engineering Education Conference (EDUCON)*, Porto, 1400–1408. IEEE.

Dichev, Christo, and Darina Dicheva. 2013. "Support for independent learning in evolving computer science disciplines." In *2013 Third World Congress on Information and Communication Technologies (WICT 2013)*, 202–207. IEEE.

Donnelly, Roisin, and Marian Fitzmaurice. 2005. "Designing Modules for Learning." In *Emerging issues in the practice of University Learning and Teaching*, Dublin: All Ireland Society for Higher Education (AISHE).

Fink, Dee. 2003. *A Self-Directed Guide to Designing Courses for Significant Learning*. San Francisco: Jossey Bass.

Garcia, André Luiz, Iury da Rocha Miguel, Jonathan Brendon Eugênio, Marina da Silva Vilela, and Guilherme Augusto Barucke Marcondes. 2020. "Scrum-Based Application for Agile Project Management." *J. Softw.* 15 (4): 106–113.

Grice, Della, and Roger Gladwin. 2004. *Developing Employability Skills*. Technical Report. Hull: LTSN Physical Sciences Centre The.

Guenzel, Lilian, and Erin Adamson Gillespie. 2020. "Agile Sales: An Exploration of Agile Methodologies in Sales." *Journal of Business, Industry and Economics* 25: 131–171.

HESA. 2017. "Destinations of Leavers from Higher Education in the United Kingdom for the academic year 2015/16." Accessed 2019-05-05. `https://www.hesa.ac.uk/news/29-06-2017/sfr245-destinations-of-leavers`.

Jahnke, Isa, and Julia Liebscher. 2020. "Three types of integrated course designs for using mobile technologies to support creativity in higher education." *Computers & Education* 146 (103782).

Joint Task Force on Computing Curricula: Association for Computing Machinery (ACM) and IEEE Computer Society. 2013. *Computer Science Curricula 2013: Curriculum Guidelines for Undergraduate Degree Programs in Computer Science*. New York, New York, USA: Association for Computing Machinery.

Mcinnis, Craig. 2000. "Changing Academic Work Roles: The everyday realities challenging quality in teaching." *Quality in Higher Education* 6 (2): 143–152.

Measey, Peter, Chris Berridge, Alex Gray, Lazaro Wolf, Peter Measey, Les Oliver, Barbara Roberts, Michael Short, and Darren Wilmshurst. 2015. "Agile foundations: principles, practices and frameworks." BCS.

Mitchell, Susan M, and Carolyn B Seaman. 2009. "A comparison of software cost, duration, and quality for waterfall vs. iterative and incremental development: A systematic review." In *2009 3rd International Symposium on Empirical Software Engineering and Measurement*, 511–515. IEEE.

Neumann, Michael, and Lars Baumann. 2021. "Agile Methods in Higher Education: Adapting and Using eduScrum with Real World Projects." In *2021 IEEE Frontiers in Education Conference (FIE)*, 1–8.

Nyborg, Mads, and Andrea Valente. 2021. "An agile approach to teach introductory programming in the hybrid classroom." 7, 40–41. Institute of Electrical and Electronics Engineers Inc.

Raiker, Andrea. 2010. "Defining reflection to enhance student attainment." *Creating Communities: developing, enhancing and sustaining learning communities across the University of Bedfordshire* 45.

Ramnath, Sarnath, and John H. Hoover. 2016. "Enhancing Engagement by Blending Rigor and Relevance." In *Proceedings of the 47th ACM Technical Symposium on Computing Science Education*, New York, feb, 108–113. ACM.

Razak, Rosnisa Abdull, and Fairul Rizal Fahrurazi. 2011. "Agile testing with Selenium." In *2011 Malaysian Conference in Software Engineering*, 217–219.

Sarikaya, Erkan, Selami Bagriyanik, and Mesut Gökalp. 2020. "Teaching Agile Software Development Using Agile Methods: A Case Study." 10. Institute of Electrical and Electronics Engineers Inc.

Schwaber, Ken, and Mike Beedle. 2002. *Agile software development with Scrum*. Vol. 1. Prentice Hall Upper Saddle River.

Shadbolt, Nigel. 2016. *Shadbolt Review of Computer Sciences Degree Accreditation and Graduate Employability*. London: Department for Business, Innovation and Skills.

Soegoto, ES. 2018. "Implementing Laravel framework website as brand image in higher-education institution." In *IOP Conference Series: Materials Science and Engineering*, Vol. 407, 012066. IOP Publishing.

Stadler, Manuel, Raoul Vallon, Martin Pazderka, and Thomas Grechenig. 2019. "Agile Distributed Software Development in Nine Central European Teams: Challenges, Benefits and Recommendations." *International Journal of Computer Science and Information Technology* 11: 01–18.

Wakeham, William. 2016. "Wakeham Review of STEM Degree Provision and Graduate Employability." Accessed 2023-08-02. `https://www.gov.uk/government/publications/stem-degree-provision-and-graduate-employability-wakeham-review`.

Young, Clive, and Nataša Perović. 2016. "Rapid and Creative Course Design: As Easy as ABC?" *Procedia - Social and Behavioral Sciences* 228: 390–395.

Zhang, Chi, Han Reichgelt, Becky Rutherfoord, Bob Brown, and Andy Ju An Wang. 2012. "Developing and improving interdisciplinary health information technology certificate programs." In *Proceedings of the 13th annual conference on Information technology education - SIGITE '12*, New York, 43–47. ACM Press.