Empirical Software Engineering (2023) 28:21
https://doi.org/10.1007/s10664-022-10252-0

Incorporating software security: using developer workshops to engage product managers

Charles Weir[1] · Ingolf Becker[2] · Lynne Blair[1]

Accepted: 20 October 2022 / Published online: 24 December 2022

Abstract

Evidence from data breach reports shows that many competent software development teams still do not implement secure, privacy-preserving software, even though techniques to do so are now well-known. A major factor causing this is simply a lack of priority and resources for security, as decided by product managers. So, how can we help developers and product managers to work together to achieve appropriate decisions on security and privacy issues? This paper explores using structured workshops to support teams of developers in engaging product managers with software security and privacy, even in the absence of security professionals. The research used the Design Based Research methodology. This paper describes and justifies our workshop design and implementation, and describes our thematic coding of both participant interviews and workshop discussions to quantify and explore the workshops' effectiveness. Based on trials in eight organizations, involving 88 developers, we found the workshops effective in helping development teams to identify, promote, and prioritize security issues with product managers. Comparisons between organizations suggested that such workshops are most effective with groups with limited security expertise, and when led by the development team leaders. We also found workshop participants needed minimal guidance to identify security threats, and a wide range of ways to promote possible security improvements.

Communicated by: Sigrid Eldh, Davide Falessi, Burak Turhan

This article belongs to the Topical Collection: *Special Issue on Software Engineering in Practice*

✉ Charles Weir
c.weir1@lancaster.ac.uk

Ingolf Becker
i.becker@ucl.ac.uk

Lynne Blair
l.blair@lancaster.ac.uk

[1] Computing and Communications, Lancaster University, Lancaster, UK

[2] Security and Crime Science, University College London, London, UK

Empowering developers and product managers in this way offers a powerful grassroots approach to improve software security worldwide.

Keywords Developer centered security · Software security · Software developer · Cybersecurity · Software development · SDLC · Product management · Product manager · Design based research

1 Introduction

Software security and privacy are now major issues: almost every day we hear that several more organizations' software systems have been compromised (RiskBased Security 2020).

While there are many aspects to an organization's security and privacy, the specification, design, and implementation of the software used has a significant impact on whether such breaches happen. Two industry trends contribute to this impact: the increasing use of microservices and Software as a Service (SaaS) components, and the DevOps movement. Both require security to be 'in the code' rather than being the responsibility of separate operations or security teams. So, development teams must be effective at creating secure software.

Unfortunately, there is evidence that developers are not delivering sufficient security. A report from Veracode concluded that *"more than 85 percent of all applications have at least one vulnerability in them; more than 13 percent of applications have at least one very high severity flaw"* (Veracode 2018). A report from Microsoft found that 28% of Software as a Service applications were not supporting data encryption (Microsoft 2018). Industry practices are not yet sufficient to support developers in providing the software security[1] we need.

In particular, it may not matter how enthusiastic a software development team may be about security. Unless they have appropriate knowledge, time and resources—both financial and otherwise—to make their software secure, they are unlikely to be effective at achieving it (Weir et al. 2019; Rauf false 2022). Yet development teams are rarely free to decide how to allocate their own time and resources. Instead, such decisions are taken by a product owner, customer, senior manager, or product management committee. This role, which we shall call 'product manager', is to ensure that the developers create the software most needed by the organization. So, how are developers to engage with product managers to achieve appropriate time and resource expenditure for the security issues in their development?

To work effectively with product managers on security makes a range of demands on the developers involved, including:

1. Understanding the relevance of security as a business driver;
2. Identifying types of security issues relevant to current projects;
3. Characterizing those issues in terms of impact and likelihood to identify the most important;
4. Identifying and costing solutions, such as security-improving activities ('assurance techniques'), to address those important issues; and
5. Discussing those issues and solutions in terms meaningful to product managers.

[1] This paper uses the words 'secure' and 'security' to include privacy aspects of the software developed, except where privacy is explicitly differentiated. 'Developers' refers to all those involved with creating software: programmers, analysts, designers, testers, and managers.

Items 2, 3 and 4 are now relatively well-understood among cybersecurity experts and some developers (Bell et al. 2017). Items 1 and 5, engaging with product managers with security as a business driver, appear less explored and understood in literature and practice.

Specifically, this paper explores outcomes from a project to create an intervention to help organizations improve the security of the code they develop, and specifically to address the five demands above. Given the vast range of types of software development, and the differences between teams in set-up, organization structure, team culture and personalities involved, it seemed unlikely we would find a 'one size fits all' method to teach to the development teams involved. Instead, we took a different approach, using 'Flipped Teaching' (Franqueira and Tunnicliffe 2015): structured activities to help participants learn from their own experience and knowledge. This took the form of a sequence of three short structured workshops to help the developers learn and identify for themselves ways to improve.

The primary research question explored by this paper, therefore, is:

> *RQ1 How can an intervention based on short workshops assist developers in identifying security issues, assessing them, and engaging product managers with those issues?*

1.1 Contribution

This paper describes the design of the three workshops and the intervention process, their use in eight different organizations, the analysis of this use, and the practical and theoretical conclusions related to engaging product managers. The research makes the following contributions:

1. It demonstrates the ability of developers to represent security enhancements in terms of their business benefits;
2. It categorizes a range of such business benefits, as identified by participating development teams;
3. It identifies factors that encourage or discourage the engagement of product managers with security ('product management engagement'); and
4. It provides an existence proof that an 'intervention package', structured as a facilitated series of workshops for a software development team, can help product management engagement.

The paper builds on an earlier paper (Weir et al. 2021a), and describes the same intervention and trials. The major additional material is as follows:

- This paper focuses on product management engagement, rather than improvements in assurance technique use, and provides new analysis to support that focus (Sections 1, 2.4, 3.5, 4.2, and 5.6);
- To address the Empirical Software Engineering readership, the full methodology is described in detail in Sections 4.3 and 4.4; and
- The paper includes the analysis of 47 hours of discussions and presentations in the workshops (Sections 4.3, 5.5, 5.6, and 5.8), to generate the following additional material:

- A discussion of security 'selling points' identified in the workshops (Sections 4.2, 5.5), and
- A discussion of factors supporting and opposing product management engagement (Section 5.8).

The rest of this paper is as follows. Section 2 discusses relevant past research; Section 3 describes the requirements for the intervention package and how they were implemented. Section 4 describes the research method and introduces research sub-questions. Section 5 explores the results from using the intervention to answer the research questions; Section 6 discusses those results; and Section 7 provides a conclusion.

2 Background

Research related to interventions and decisions for secure software has taken a variety of disparate approaches. In this section, we explore how research has explored security-oriented interventions and the relationship with product managers. Specifically, we discuss ways to get developers to adopt business process improvements related to security; consultancy and training interventions; approaches to motivate developers towards security; blockers and motivators as a means of analysis; and work studying how product managers engage with developers on security.

2.1 Adoption of developer security activities

One way to incorporate development security into organizational practice is to build a process around it using a 'Secure Development Lifecycle' (SDL). This is a prescriptive set of instructions to managers, developers and stakeholders on how to add security activities to the development process (De Win et al. 2009). However, research suggests resistance from development teams to adopting a prescriptive methodology. For example, Conradi and Dybå (2001) deduced in a survey that developers are skeptical about adopting the formal routines found in traditional quality systems.

van der Linden et al. (2020) found from a task-based study and survey that developers tend to see only the activity of writing code to be security-relevant. They suggested a need for a stronger focus on the tasks and activities surrounding coding. And an interview survey by Xie et al. (2011) suggests that developers make security errors by treating security as *"someone else's problem,"* rather than as a process involving themselves.

Moving on to security-promoting interventions, Türpe et al. (2016) explored the effect of a single penetration testing session and workshop on 37 members of a large geographically-dispersed project. The results were not encouraging; the main reason was that the workshop consultant highlighted problems without offering much in the way of solutions. A study by Poller et al. (2017) followed an unsuccessful attempt *"to challenge and teach [the developers] about security issues of their product"*. The authors found that pressure to add functionality meant that attention was not given to security issues and that normal work procedures did not support security goals. They concluded that successful interventions would need *"to investigate the potential business value of security, thus making it a more tangible development goal"*.

Other work has also found a need for the business alignment of software security. Caputo et al. (2016) concluded from three case studies a need for the alignment of security goals with business goals. Weir et al. (2020b) surveyed security specialists working with developers,

identifying a frequently-used approach for developer teams of 'product negotiation': involving product managers and other stakeholders in security discussions.

Considering solutions to support developers, Yskout et al. (2015) tested if 'security patterns' might be an effective intervention to improve secure development in teams of student software developers. The results suggested a benefit but were statistically inconclusive. Such et al. (2016) defined a taxonomy of twenty assurance techniques from a survey of security specialists, finding wide variations in the perceived cost-effectiveness of each. And a recent book by Bell et al. (2017) provides support for developers and tool recommendations, containing much valuable practitioner experience, but little objective assessment of the advice provided.

2.2 Motivating change in development teams

Dybå (2005) concluded from a quantitative survey that organizational factors were at least as important as technical ones to motivate change in development teams. They found that actions need to be aligned with business goals, and a need for employees to take responsibility for the changes. Beecham et al. (2008) conducted a literature review of 92 papers on programmer motivation in 2008, concluding that professional programmers are motivated most by problem-solving, by working to benefit others and by technical challenges. Hall et al. (2008) framed these motivators as 'intrinsic', relating them to self-determination theory (Herzberg 2017).

Lopez et al. (2019a) concluded that to encourage developer security there is a need to *"raise developers' security awareness;"* they successfully used 'playful workshops' to do so (Lopez et al. 2019b).

More generally, awareness is just the first step (Beyer et al. 2015), and individuals need to be supported through training to have the ability to perform the expected behavior (Fogg 2009). Organizations need to integrate security tasks into the primary business activities, rather than 'bolting them on' afterwards through unworkable policies or compliance exercises (Kirlappos et al. 2013).

2.3 Blockers and motivators

Apart from raising awareness of the importance of security, the workplace environment, individual rewards and perceived potential negative consequences are important factors affecting developers' adoption of secure practices (Assal and Chiasson 2019). Pfleeger et al. (2014) observed that the key to enabling good security behavior is good 'motivators': feedback, situations or rewards that encourage the behavior. But piling on motivations is not sufficient. If individuals are faced with obstacles—'blockers'—these need to be removed before the desired behavior can be achieved (Tietjen and Myers 1998). Furthermore, individuals may feel that they are 'unequipped for security' or, potentially even worse, disillusioned to the benefit of promoting security. In that case, motivators will be perceived as a nuisance and may reinforce archetypal behaviors (Becker et al. 2017; Assal and Chiasson 2019).

2.4 Product management engagement

While there is an extensive literature on methods for secure requirements engineering (Nhlabatsi et al. 2012), there is less work investigating how the need for such requirements

is established and motivated: Ambreen et al. (2018) found only 16 papers discussing the practical effects of requirements engineering out of a total of 270 dedicated to empirical requirements engineering. Typically these were case studies of the application of specific approaches (Mead and Stehney 2005; Mellado et al. 2006). Much of the product manager role is one of prioritization: research has developed several technical approaches to prioritization (Bukhsh et al. 2020), some of which prioritize non-functional requirements including security against functional ones (Dabbagh et al. 2016); however, we found no evidence in the literature that software product managers have used them in practice.

Exploring product management more generally, Springer and Miler (2018) identify 8 personas and an archetype for software product managers; they note that many started in development roles. Standard texts for product managers tend to explore practical decision-making within the role, e.g. (Haines 2014). We have found no other empirical research studying the interaction related to security between developers and product managers.

Much work has been done supporting development teams and product managers with the wider scope of non-functional requirements, of which security can be regarded as one. SEI's Quality Attribute Workshop, for example, brings together developers, product managers and other stakeholders to identify and quantify such non-functional requirements (Barbacci et al. 2000); it addresses security through 'quality attribute scenarios'. Though powerful it requires considerable effort and the participation of a wide range of stakeholders.

2.5 Conclusions

This previous work suggests a need for lightweight interventions to improve the interaction between developers and product managers to support better engagement in security. In particular, we observe in Section 2.1 a need to align developers' security goals with business goals.

3 Design of the intervention workshops

This section explores the design criteria and creation approach for the intervention. We expressed the design criteria in terms of 'Requirements', using the term in the requirements engineering sense to mean the explicit and implicit needs and wants of the stakeholders using the intervention (Nhlabatsi et al. 2012). As discussed in Section 1, we wanted an intervention to help developers in:

> Requirement 1 Understanding security decisions as business decisions;
> Requirement 2 Identifying types of security issues relevant to their current projects;
> Requirement 3 Characterizing those issues in terms of their importance to the organization;
> Requirement 4 Identifying and costing solutions to address the important issues; and
> Requirement 5 Discussing those issues and solutions in terms meaningful to product managers.

We also identified, based on industry experience and previous literature, several further implicit requirements for such an intervention, specifically that it should:

Requirement 6 Take less than one working day for a development team to carry out, to keep costs acceptable;
Requirement 7 Work with development teams, as a majority of developers work in teams (Stack Overflow 2016);
Requirement 8 Work without security specialists, since many teams do not have access to them (Weir et al. 2020a);
Requirement 9 Work without product managers present in the workshops, since while it is obviously a benefit to include them, in many cases they may not be available or persuaded to attend;
Requirement 10 Support developers currently using few or no assurance techniques, since many teams do not currently use them (Weir et al. 2020a); and
Requirement 11 Be leadable by non-researchers, to permit the use of the intervention where the researchers are unavailable (Weir et al. 2019).

The following sections explore the implementation of the each of the above requirements in turn.

3.1 Requirement 1: Understanding security in terms of business decisions

To help developers understand decision making around security we used a facilitated game, the 'Agile App Security Game' based on the game 'Decisions Disruptions' (Frey et al. 2017), which is now used extensively in the UK in management cybersecurity training (Shreeve et al. 2020). In it, the participants work in groups as product managers, discussing and selecting security-enhancing product improvements with varying costs and learning whether their choices deter attacks. The Agile App Security Game uses a different case study project from Decisions Disruptions, with developer-oriented threats and mitigations that have been updated over several years. The game has two implied lessons for the participants:

- There is no need to have a security expert present to make decisions about software security (Requirement 8)
- Winning, by defending against every threat, is virtually impossible. It is a business decision as to which threats to address, based on which ones are most important to the organization.

3.2 Requirement 2: Security issues relevant to current projects

The activity of identifying specific kinds of security issues for a given project is an important assurance technique for security (Such et al. 2016). This activity, which we term 'threat assessment', was challenging to teach and implement in a short workshop. Though valuable, standard 'threat modeling' approaches require considerable knowledge of possible technical threats, and preferably support from a professional with a detailed understanding of both the industry sector and current cyber threats to it (Shostack 2014); we could not assume either would be available.

It seemed possible that developers might require classroom training in threat modeling techniques. In creating the workshops, though, we instead followed the agile practice of

trialing the 'simplest thing that could possibly work' (Beck and Fowler 2001). So, as an experiment, we hypothesized that developers would need no training.

We, therefore, used a lightweight threat assessment approach, specifically a facilitated ideation session (Fisher et al. 2011). The participants were asked to address the open question: "*Who might do what bad thing to whom?"* in the context of their current project. In all but the last workshop, all the participants faced a flipchart, and a facilitator wrote down unfiltered suggestions. One group (Group K) were particularly expert at facilitation. In their workshop, participants discussed the question in groups of about six, creating post-it notes with suggestions, and placing them on a shared whiteboard.[2]

3.3 Requirement 3: Issues in terms of impact and likelihood

To make decisions about threats, Requirement 3 was to characterize each type of threat in terms of its importance to the organization. We approached this using the standard risk management approach of estimating the likelihood and impact for each threat. To do this rigorously requires considerable knowledge of the business environment, of current trends in cybersecurity and of risk management theory and practice (Hubbard and Seiersen 2016).

For the workshops, however, we needed only to introduce the concepts in the simplest way that could add value for the participants. So, as part of the Threat Assessment workshop, participants used 'dot voting' to decide likelihood and impact information. Each of the participants used a set of 3 red and 3 black colored dots to vote on the most likely and most impactful types of threat. Based on the votes, the workshop facilitators organized the types of threat into an ad-hoc 3 × 3 Risk-Impact grid. Figure 1 shows an example.[3] This then enabled participants to select a set of the four or so 'most important issues'.

3.4 Requirement 4: Identifying and costing solutions

Identifying and estimating costs for solutions to these most important issues was similar to other development tasks, and therefore a skill the participants had already (Requirement 4). To keep the workshops short (Requirement 6), the workshop involved only a superficial solution and costing in each case. We did, however, identify that it was important to remind or teach developers standard approaches to improving security (Requirement 10). We approached this by encouraging the facilitator to discuss, wherever relevant, a small set of assurance techniques: configuration review, automated static analysis, source code review, and penetration testing (Such et al. 2016).

3.5 Requirement 5: Discussing in terms meaningful to product managers

From prior literature and earlier work of our own, we had identified that product managers had difficulties engaging with messages along the lines of "*we must do this security enhancement or terrible things will happen.*" This reflects two problems: (1) where a 'bad' decision has a large cost, it can often lead to 'analysis paralysis' (Haines 2014, ch 5); and (2) our observation that it is difficult for product managers to compare positive improvements, such as new features, against risks of negative consequences.

[2] All three elements of this approach have been adopted in the current version of the workshop package.
[3] The post-it colors have no significance; the post-it text is deliberately blurred.

Fig 1 Whiteboard with Risk-Impact Grid

To address these problems (Requirement 5), we hypothesized that it might be better to explore with product managers the *benefits* of addressing specific security issues (Ashenden and Lawrence 2013). Therefore, as an experiment, we added a further 'Security Promotion' workshop. In this workshop, developers identified ways to represent the solutions to their identified threats as positive enhancements: presenting security as a positive good (McSweeney 1999). While it may be helpful to have product managers present in this workshop to represent the 'product manager point of view', it was by no means necessary (Requirement 9).

As in the identification of threats (Section 3.2), we had originally thought that developers might require classroom training in techniques to do this. In creating the Security Promotion workshop, though, we again followed agile practice by trialing the 'simplest thing that could possibly work' and omitting any training. Participants split into groups, and each group addressed one of the threats from the most important five or so identified in the threat assessment. The instruction for the participants was to *"work out positive ways in which addressing that threat will benefit the organization"*. Each group discussed the threat they had chosen and wrote notes on a whiteboard or flipchart page. A representative from each group then presented their conclusions to the other participants. Following these presentations, the participants decided on project actions to carry out after the workshops.

3.6 Remaining requirements

The remaining, implicit, requirements were addressed as follows. To address Requirement 6 (less than one working day), we limited the work identifying and costing mitigations as described in Section 3.4. For Requirement 7 (working with teams) we had teams of developers attend the workshops and discuss their own projects there. For Requirement 8 (avoiding security specialists) and Requirement 10 (for developers using few assurance techniques) we kept discussions and outputs away from technical security knowledge and activities. To address Requirement 9, the workshops did not rely on any product manager involvement.

To address Requirement 11 (leadable by non-researchers), we trained one or two facilitators from each organization, and they then managed the intervention. The training was a 1–2-hour interactive face-to-face discussion, ('Facilitator Training'). Here, we discussed the role of the facilitator in each workshop in turn, including points for them to emphasize and possible pitfalls. We provided the facilitators with materials (Weir et al. 2021b) to give the workshops: cards and instruction sheets for the game; and PowerPoint slides with participant instructions for the subsequent workshops.

3.7 Intervention approach and schedule

We recruited one or more development teams (a 'group') in each of eight organizations and carried out the intervention with them. With each group, we first interviewed a selection of the participants to establish a baseline in terms of their current understanding, practice, and plans ('before' interviews). We then trained the facilitators, who led the intervention workshops. To track the effects of the intervention, we held two monthly follow-up sessions, typically hour-long video conferences, between the researchers and participants. Finally, about three months after the start we carried out 'after interviews' with the same participants as before. Both 'before' interviews and 'after' interviews were semi-structured using open questions; Appendix A lists the questions used; these were as used in an earlier project (Weir et al. 2019).

Researchers attended all the workshop sessions, recording the audio of the participant discussions for later analysis. Author Charles Weir acted as main intervener; author Ingolf Becker supported work with Group K.

Figure 2 shows a typical schedule for delivering the interventions, distinguishing the different sets of participants in each activity. As shown, where possible the three workshops—Agile App Security Game, Threat Assessment, and Security Promotion—were all held on the same day, along with the 'before' interviews and the facilitator training, using approximately the timings shown; for some groups they were held over two consecutive days. The 'after' interviews were with the same subset of the participants as the 'before' interviews; the subset that attended the follow-up sessions varied between companies. The research engagement with each group spanned 3–4 months, with researchers on-site for only one to two days at the start and a day at the end. As shown, the combined time for the three workshops (items labeled A) was about 5 hours, satisfying Requirement 6 of *taking less than a day*. The overall involvement time was limited to four months to provide long enough to achieve change, but not so long that impact could become difficult to distinguish from other influences.

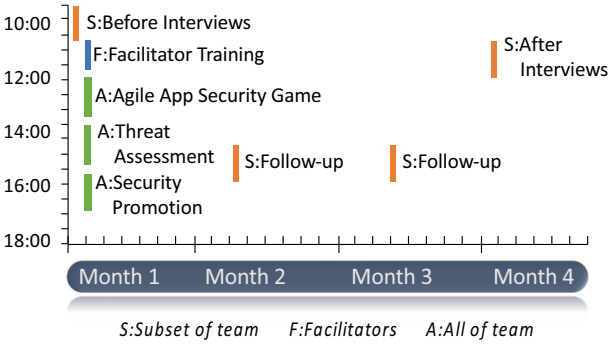

Fig. 2 Typical Intervention Timeline

4 Evaluation methodology

Our approach to the research was pragmatic: we wanted to achieve an effective intervention that could help a large number of software developers (Easterbrook et al. 2008). We chose Design-Based Research (DBR) as our methodology for the project for the following main reasons: DBR focusses on designing an artifact, accepts the involvement of researchers in trials, develops both academic theory and practical outcomes, has a cyclical approach, and supports different users for the artifact in each cycle (Kelly et al. 2008). We considered other methodologies. One, Action Research requires following the same participants through multiple cycles of intervention, but in this project, participants changed between trials of the intervention. Another methodology, ethnography, requires the researchers to take a passive role. Most other approaches require non-intervention by the research team. DBR provided the best 'fit' to the research.

4.1 Introduction to design-based research

DBR has its roots, and is used most, in education research. Its foundation lies in the 'design experiments' of Brown (1992), and Collins (1992) working with teachers as co-experimenters. It emphasizes the development of design theory in parallel with the creation of teaching innovations. DBR is now an accepted research paradigm, used to develop improvements ranging from tools to curricula (Design-Based Research Collective 2003), with a recent guide book for practitioners (Bakker 2018).

The characteristics of DBR are that it is: '*pragmatic*', aiming to solve real-world problems by creating and trialing interventions in parallel with the creation of theory; *'grounded'* in the practicalities of real-world trials in the *"buzzing, blooming confusion of real-life settings"* (Barab and Squire 2004); '*interactive', 'iterative'* and *'flexible'* with an iterative process involving multiple trials and experiments taking place as the theory develops; '*integrative'* in that DBR practitioners may integrate multiple methods, and vary these over time; and '*contextual'* in that results depend on the context of the real-world trials (Wang and Hannafin 2005). Figure 3, based on Ejersbo et al. (2008), shows the two parallel cycles of DBR research: creating theory and creating the artifact. The bold, colored, arrows are additions based on the authors' own experience of the DBR process.

The practical aspects of carrying out DBR are defined by the 'integrative' nature of DBR: both design and assessment techniques must come from other research methodologies (Wang

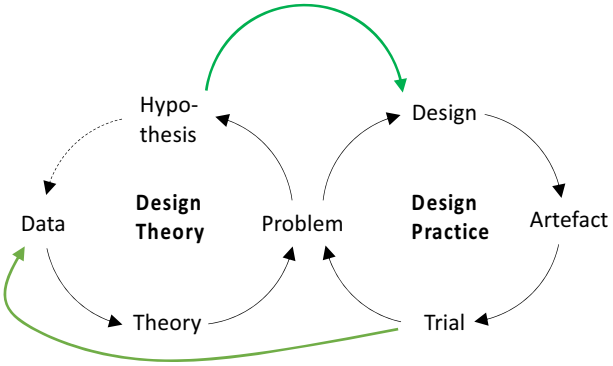

Fig. 3 Activities in Practical Design-Based Research

and Hannafin 2005). In this research, we used the *techniques* of the Canonical Action Research method (Davison et al. 2004), though not that method's overriding *paradigm*. Specifically, we participated in an intervention to help the participants change their behavior; we recorded the discussions involved, transcribed them, and analyzed them in detail; and we are using the research findings to inform changes to the intervention to incorporate into a further cycle of development.

4.2 Research questions

DBR requires separate research questions for the Design Practice cycle and the Design Theory cycle. Design Practice questions assess the qualities and effectiveness of the artifact being designed (in this case, the workshop package). Design Theory questions address the context of artifact usage, with results that can apply to other research, such as the creation of different interventions. Accordingly, we need to break down the primary research question RQ 1 (*How can an intervention based on short workshops assist developers in identifying security issues, assessing them, and engaging product managers with those issues?*) into sub-questions: specifically, Design Practice questions, and Design Theory ones.

Our first Design Practice question explores the workshops' overall impact:

> *RQ 1.1 To what extent did the developer teams achieve better product management engagement over security issues as a result of the intervention?*

The second Design Practice question considers the outcomes of the Security Promotion workshop, since these outcomes may be of value for other teams in future:

> *RQ 1.2 What did participants identify as 'selling points' for improvements in software security?*

For this purpose, we used a standard definition of a selling point, as *a feature of a product for sale that makes it attractive to customers (*Oxford Languages 2011*).*

And another question explores differences between the results in different organizations, to indicate how widely applicable the intervention may be:

RQ 1.3 In what ways do the intervention results vary with different participant contexts?

Turning to Design Theory questions, the hypothesis that presenting a positive view of security would help engagement (Section 3.5) was speculative, and needed testing:

RQ 1.4 Can having developers consider the positive benefits of security and privacy mitigations lead to improvements in product management engagement?

In creating the workshops, we had hypothesized that developers would require no training to carry out the activities in the Threat Assessment and Security Promotion workshops (Sections 3.2, 3.5). We, therefore, posed this further research question to test this hypothesis:

RQ 1.5 Can teams of developers produce threat assessments, risk-impact assessments, and benefit analyses with minimal guidance?

Finally, to help explore the 'how' of RQ 1 (*How can an intervention ... assist developers...*) we wanted to identify any other aspects related to product management engagement that might help to explain the working of interventions aiming to help improve developer security practice:

RQ 1.6 What are the 'blockers' and 'motivators' affecting product management engagement and other stakeholders as revealed in the workshops?

For this question, we define blockers to be factors that prevented engagement or made it more difficult; motivators are correspondingly those factors that encourage such engagement.

4.3 Method implementation

We recorded the audio of all the interviews and all the workshops for each group, then transcribed the interview audio manually, and the workshop audio using an automated transcription service.[4]

To evaluate the Design Practice question RQ 1.1, *To what extent did the developer teams achieve better product management engagement over security issues as a result of the intervention?,* our approach was as follows. Two authors coded the interview transcripts in an iterative process, using NVivo. We used the techniques of Thematic Analysis (Clarke et al. 2015), coding statements in the 'before' and 'after' interviews that referred to one of the two 'activities' related to the question shown in Table 1.

We also coded, for the same statements, corresponding Adoption Levels that the participants in each group might achieve for each activity, as shown in Table 2.

During the coding, we were particularly careful to distinguish changes due to the interventions from those due to other external factors; we did not code the latter.

To assess the impact (security improvement resulting from the intervention) we extracted, for each group and each coder, the highest recorded Adoption Level for each activity, both before and after the intervention. Initially, both coders coded one group's interviews independently, then met to discuss differences and agree on interpretations going forward. We both

[4] https://sonix.ai/

Table 1 Activities Analyzed

Activity	Description
Threat assessment	Design-level analysis of possible attackers, motives, and vulnerability locations.
Product management engagement	Working with product managers to make security decisions.

then coded all the interviews and calculated an initial Inter-Rater Reliability based on that coding. We met to discuss the differences, then independently recoded all the interviews and calculated a final Inter-Rater Reliability figure. Our Inter-Rater Reliability calculations used Krippendorff's Alpha (Gwet 2014) to compare the Adoption Levels calculated from the coding of each coder. See Section 4.4 for an illustrative example.

To combine the ratings of the two coders, we took the highest Adoption Level recorded by either coder. Given we were studying changes in Adoption Levels, to avoid bias we needed only that the combination method be consistent across 'before' and 'after' interviews. See Section 5.2 for the practical justification for using the highest values. Using the numerical rating of each Adoption Level as shown in Table 2, we calculated the 'impact' of the intervention on the participants' adoption of each activity, as the difference between the Adoption Level in the 'before' interviews and the Adoption Level in the 'after' ones. Of course, this impact calculation is merely an indication. For example, a two-unit change in Adoption Level might be from '0 No Mention' to '2 Planned', or from '2 Planned' to '4 Established'; these changes are not semantically equivalent.

To explore question RQ 1.1 further, we later looked in detail at the nature of each improvement and identified and extracted exemplar quotations from the interviews.

For RQ 1.2 (selling points), a single researcher coded all the workshop and training session audio using closed Thematic Analysis (Clarke et al. 2015). The automated transcription quality was poor, as expected, so the researcher coded from the audio, using the transcripts only for easier navigation and as placeholders for the codes. Aspects coded included 'blockers', 'motivators', and 'selling points identified'. To further address RQ 1.2, a single researcher extracted the text coded as 'selling points' and used open Thematic Analysis (Clarke et al. 2015) to further categorize kinds of selling points.

To explore RQ 1.3 (variation with context), we calculated how the impact varied with different attributes of the participants from each group: the organization size, facilitation style, team security maturity, whether a product manager was present, and the job description of the lead facilitator. To do this, we calculated the mean impact for each activity for different values of each attribute. Again, since impact values are not semantically consistent, this mean cannot be used for comparing results for different activities against each other, but it does allow us to identify where changes in Adoption Level tended to occur most.

Table 2 Adoption Levels for each Activity

Level	Description
0 No mention	No reference to it in the interview
1 Aware	The team showed knowledge of it.
2 Planned	Existing plans to incorporate it.
3 Using	The team have used it.
4 Established	The team use it in each new project.

Table 3 Illustration of Adoption Level values for Group D's Interviews

Code	Before:			After:			Impact
	Rater1	Rater2	Combined	Rater1	Rater2	Combined	
Product management engagement	0	0	0	4	4	4	4
Threat assessment	1	1	1	3	4	4	3

For RQ 1.4 (positive benefits improving product management engagement), we considered the answers to RQ 1.2, along with the impact assessment of the product management engagement.

We addressed RQ 1.5 (unsupported threat assessments) with the analysis described for RQ 1.2 above. Additionally, we reviewed the discussions that took place in the three workshops as well as the outputs produced.

For RQ 1.6 (blockers and motivators), we used the same analysis as RQ 1.2 and RQ 1.5 above. We then categorized the blockers and motivators identified, using open Thematic Analysis to provide a basis for their description.

The calculations and graphics creation used the qualitative data analysis tool NVivo,[5] Microsoft Excel, and Python in Jupyter Notebooks (Kluyver et al. 2016). The research was approved by the Lancaster University Faculty of Science and Technology Research Ethics committee.

All the quotations from the recordings in this paper were manually transcribed and checked for correctness.

4.4 Example of the impact coding

Figure 4 illustrates the impact calculation used for RQ 1.1 and RQ 1.3, showing the final coding for an 'after' interview from Group D. In it both coders identify a statement indicating the adoption of threat assessment, but the coders disagreed on the level of adoption implied. So, two different Adoption Levels would be extracted: "*D – After – threat assessment: 3 Action*" for coder Rater1 and "*D – After – threat assessment: 4 Incorporation*" for coder Rater2.

Table 3 shows an illustrative set of extracted values based on Fig. 4. The Krippendorf's Alpha Inter-Rater Reliability calculation would be based on both sets of columns Rater1 and Rater2 in that table.

The 'Combined' columns in Table 3 shows the highest Adoption Level recorded by either coder. From them, the table calculates the product management engagement impact for D as $4 - 0 = 4$, and the threat assessment impact as $4 - 1 = 3$.

[5] https://www.qsrinternational.com/nvivo-qualitative-data-analysis-software/home

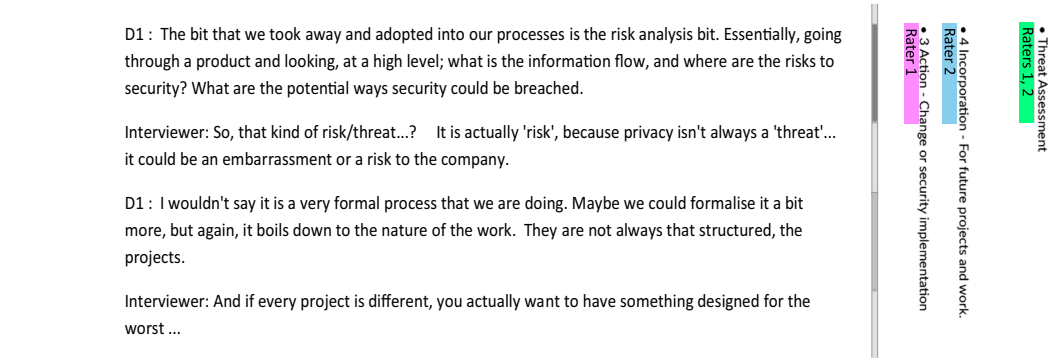

Fig. 4 Example Coding from A D 'After' Interview

5 Results

This section explores the results from the project, and addresses each of the Design-Based Research questions RQ 1.1 through RQ 1.6.

The intervention was carried out with a total of 88 developers in eight different organizations, generating 21 hours of interview audio, and 47 hours of audio from training, workshop, and follow-up sessions. The final code book contained 2859 references to 51 codes. Practical considerations and technical issues meant that not every workshop and team discussion was recorded. However, all the important points discussed in the non-recorded events were covered in interviews or other workshops in sufficient detail not to impact the quality of our data.

5.1 Summary of participants from each organization

The participant organizations were recruited opportunistically through industry contacts, university outreach and software developer conferences. Table 4 describes the organizations and groups involved. Organizations are identified with a letter, starting with D (since three organizations had been involved in earlier trials. All the developers interviewed were male, as were all the team line managers and quality assurance specialists; three product managers were female. These numbers are consistent with industry norms (Stack Overflow 2016).

Figure 5 visualizes the participants. It plots the organization sizes (ranging from F's 20 staff to E's 6000 staff), against an estimate of their 'secure software capability maturity' (ISO/IEC 2008) based on the participants' discussions during the workshops. Ring sizes show the number of participants (3 in F to 16 in K); ring centers show the facilitators; colors and hatching distinguish the job roles.

5.2 Inter-rater reliability

The Krippendorff's Alpha Inter-Rater Reliability calculation on the adoption levels of activities after the first round of coding[6] (Sections 4.3, 4.4) was 0.18, indicating only slight agreement (Viera and Garrett 2005). The main cause of disagreement was that the interviewees had not been asked

[6] This initial IRR calculation applied to 12 different activities (Weir et al. 2021a); this paper describes only two of them.

Table 4 Description of Participants

	Organization	Group
D	A development team within a university, funded by a government grant to promote business innovation by developing proof of concept (PoC) applications.	Aware of the importance of software security but had little practical knowledge; worked on several different projects at a time.
E	A government department delivering software for sensitive government applications. The group worked on a high-confidentiality product.	Less experienced than average for the industry, though the session leader is an experienced security specialist.
F	A small surveying company delivering a Geographical Information System product and related services.	A previous developer had implemented some security aspects; the current team had little knowledge.
G	A web applications development company delivering a wide variety of applications for clients.	The two leads were expert in software security, but were finding that the effort costs of security were not being included in client pricing.
H	A small company selling a range of Internet of Things devices and their associated infrastructure.	The group justifiably consider themselves good at software security.
I	A well-established company providing the infrastructure for a commodity trading. Planning move from perimeter security to cloud-based services.	The company has considerable internal expertise in security. However, the developers were less experienced.
J	A well-established large company providing web interfaces for retailers. The group involved had the responsibility of creating tools and services to support deployment.	The group was a team of about a dozen developers creating deployment tools, and included two security specialists who led the workshops.
K	A well-established company with a few hundred employees creating tools for developers.	The group has a strong emphasis on agile development processes, and team interaction. All the participants were developers.

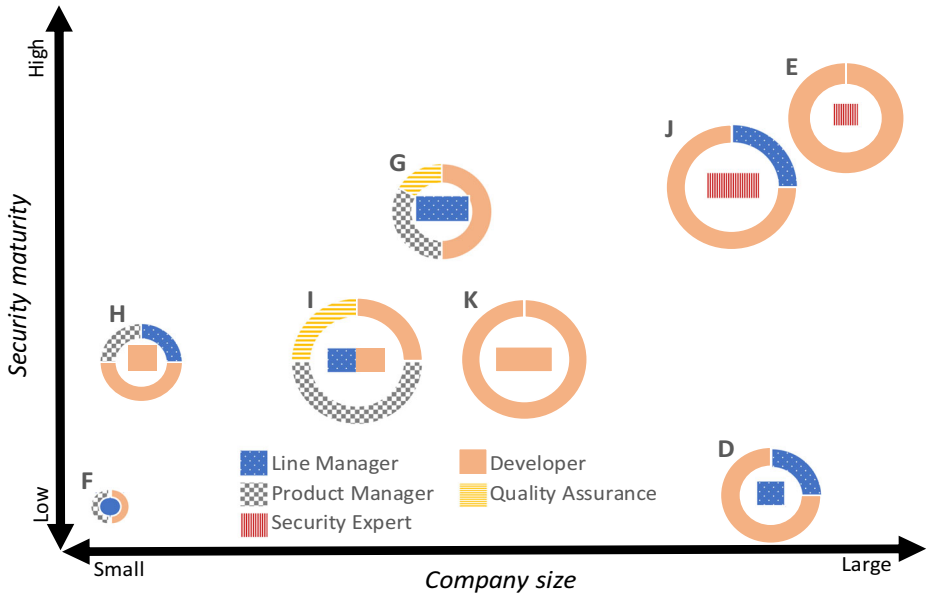

Fig. 5 Composition of the Participating Groups Circles Show Participants; Centre Rectangles Show Facilitators

explicitly about their use of the activities, in order to avoid bias in the responses. This caused several kinds of discrepancy between the interpretations of the two coders.

Once the coders had discussed the discrepancies and independently recoded the interviews, the resulting Krippendorff's Alpha metric was 0.46, indicating moderate agreement. The metric calculated for the two activities described in this paper was 0.80, indicating substantial agreement. We analyzed the remaining discrepancies and found them to be mainly omissions by one or another coder, which were mitigated by the policy of using the highest Adoption Level from each coder for subsequent analysis.

5.3 Impact of the intervention

Figure 6 summarizes the impact outcomes related to product manager Engagement, calculated as described in Section 4.3. The size of each bubble indicates the final Adoption Level for the two aspects after the intervention. The bubble's color and texture show the impact attributed to the intervention: hatched amber for a change of 1 to 2 Adoption Levels; dotted red for 3 to 4 Adoption Levels. Note that other aspects of some groups' security practice, such as the use of automated static analysis tools, also improved as a result of the intervention (Weir et al. 2021a), but those improvements are out of scope for this paper.

Figure 6 thus provides an answer to the Design Practice question RQ 1.1 *(To what extent did the developer teams achieve better product management engagement over security issues as a result of the intervention?)* Specifically, the intervention led to notably improved product management engagement in four of the eight groups involved (D, E, F and I), and led to some improvement in two further groups (G and K).

As shown, the intervention also improved understanding and use of threat assessment (*Design-level analysis of possible attackers, motives, and vulnerability locations*). This is vital to ensure that the team is as effective as possible by prioritizing the most important security issues. Six of the eight groups (D, F, G, H, I and K) were not doing this prior to the workshops; six of the eight groups (D, E, F, G, I, J) ended up using this in their current projects; one (D) adopted it as part of their process for all projects. So, for four groups (D, F, G, I) this represented a major improvement on their practice before the intervention.

Table 5 explores the detailed outcomes the outcomes in improved product management engagement as a result of the intervention. The Δ column shows the impact, using the same highlighting as Fig. 6, with quotations from the exit interviews or (in the case of Group K) workshops.

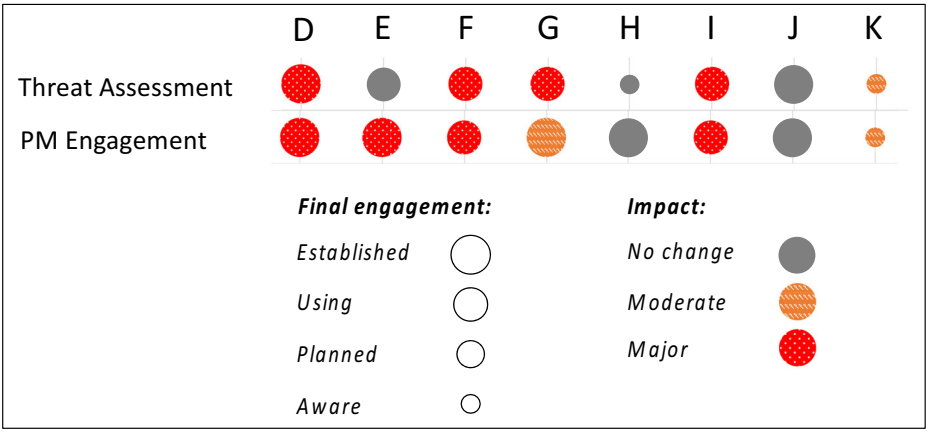

Fig. 6 Impact Related to Product Management Engagement

Table 5 Product management engagement Outcomes

	Δ	Product management engagement outcome	Quotes
D		Identified that the threat and risk assessment itself was a valuable asset to their clients, and a need for security support when the clients came to implement applications based on the PoCs.	*After the workshop…, we redesigned our handover template, which is where we now have a specific section for security*
E		Realized that while every security enhancement was essential, the ordering of their implementation could be altered to suit the client.	*So, once we were given our requirements we went away and looked at the security things, and talked to our customers about, given the time… and the requirements..., we recommend putting in this, this and this, and we didn't get them all straight away*
F		'Lined up' security improvements to be incorporated in the enhancements when new clients wanted them, and subsequently did incorporate the improvements.	*Yes, we are in a promising looking situation at the moment… we have picked up some new contracts…, so they will require us to implement pretty much everything that we had listed*
G		Agreed and adopted an impressively simple way to discuss security cost-benefit with a client: gold level hosting, silver, and bronze security.	*[A team member] did a lot of the leg work and set up gold, silver and bronze packages to say 'right, answer these 10 questions', and then you would get a points score, and "you fit in within this bracket, and this is the package that you need"*
H		Identified that their security story was a major Unique Selling Point against competitors.	*It is used as a sales thing; in that they can say "it is secure"*
I		Subsequently included security requirements in discussions with new clients. Rolled out the workshops independently of the researchers to other teams.	*When a customer asks me, "why aren't you doing this thing', I think I am in a much better position to feel that I can honestly say: yeah, … we will do something about it ... or no, we don't need to do anything about that because it is not actually that big a risk" (Product Owner).*
J		Devised several functionality and process improvements for their 'customers': development teams in other parts of the organization.	*In terms of risk assessment, now we have a new Product Owner on our team, he is quite keen to incorporate it, and also the team is quite keen. We are [also] trying to assess the impact....*
K		Each of four subgroups delivered a convincing sales pitch for a client-visible security improvement.	*The good thing of this is we [will be] the Gold Standard Security as well as everything else. For sales people this will build [customer] confidence.*

All of the groups remained relatively consistent in staff and projects during the three months of our research involvement. The monthly follow-up session (Section 3.7) meant that we could track any important changes in their customer requirements and their other security initiatives. We used these to filter out effects not due to the workshops in the analysis, as indicated in Section 4.3. We can therefore be reasonably sure that the outcomes in Table 5 are the effects of the workshops.

5.4 Activity impact by group attributes

Table 6 addresses RQ 1.3 showing how the *Impact varied with different attributes of participants* by calculating the mean impact for each activity resulting from interventions on participant teams with that attribute. The deeper shadings show the higher values in each categorization. The table shows the two activities, while the figures on the 'Overall' line show the average increment over both activities for each category. We observe that:

Less security-expert groups benefitted most from the workshops Specifically, those with a low security maturity showed the highest impact.

Table 6 Impact Averaged by Group Attributes

Categorization	Category	Count in category	Threat Assessment Impact	PM Engagement Impact
Overall		8	1.6	2.1
Organization Size	Large	3	1.	2.7
	Medium	3	2.3	2.
	Small	2	1.5	1.5
Facilitation Style	Dominating	2		2.
	Listening	4	2.3	2.3
	Peer	2	2.	2.
Security Maturity	High	2		2.
	Medium	4	1.8	1.5
	Low	2	3.	3.5
Product manager	Yes	4	2.3	2.
	No	4	1.	2.3
Lead facilitator	Line manager	4	3.	3.
	Security	2		2.
	Developer	2	0.5	0.5

Sessions facilitated by line managers were more effective than those facilitated by developers or security specialists We speculate from our observations in the workshops that this may reflect better training in facilitation-related skills for managers; it may also reflect greater power among managers to introduce new techniques.

The presence or absence of a product manager in the group had negligible effect on product management engagement This was a surprise. We had expected a product manager would encourage emphasis and therefore improvements, but the results do not show that effect.

5.5 Selling points for security

To address Design Practice question RQ 1.2, we coded all the recorded audio from interviews, workshops and training sessions for *selling points for software security.* We then used open coding to categorize each item (see Section 4.3). 50 items were found, from 20 different sessions, making a total of 4292 words.

Table 7 summarizes the findings. Each line names a category, shows the groups that identified selling points in that category and the number identified; and describes each one with quotations from the discussions[7]. Four selling points amounted to a naïve 'security is good for customers' and are omitted from the categorization.

Thus, the answer to RQ 1.2 is that professional developers can identify a large range of selling points for software security, in a variety of categories.

5.6 Use of selling points to engage product managers

To address the Design Theory research question RQ 1.4 (*Can having developers consider the positive benefits of security and privacy mitigations lead to improvements in product*

[7] Note that for Group D the Security Promotion workshop was not recorded; the single selling point was in an exit interview.

Table 7 Selling Points Identified

Name	Org.	N.	Description	Example Quotes
Security Consultancy	D E F G I	10	Being the experts in security; advising the customer; saying 'No' to feature requests that compromise security.	*The more projects we do the better we'll get at these things to the point that the security consultancy ends up being part of the package (D) Actually, [security] is not about [us] making the money; it is about making the right money for the client (G)*
Security Management	G H I J K	8	Managing security as a continuous service for customers	*What you want in a supplier is ... they're proactive in considering the [security] challenges and they're doing things about it (H)*
Customer Tick-box Requirement	F H I	7	Improvements to satisfy standard customer requirements.	*People ask if we are ISO 27001 certified (I) Got to have two factor authentication, because that's what [the customer] does with other systems (F)*
Customer Choice	E F G	6	Customer gets value by specifying level of security or order of delivery.	*We can sell tiers... this is a basic [security] package; this is our premium package. (G)* *[Sometimes] they have said 'we are happy to accept that level of risk', but there is also quite a willingness to fix a lot of the other issues. (E)*
Robust System	E H J K	6	The system will have high availability.	*Being proud of ... your availability: X nines. We have a track record: 12 months... something to talk about (J).*
Better Security than Competition	H I	4	Customers will choose this product because it has better security.	*Using [security] as a differentiator from Chinese manufacturers that can build stuff for a fraction of the cost, but wouldn't necessarily have considered the bigger picture (H)*
Implied Requirement	E F K	3	Security enables a new item of functionality.	*They've said, "could you put in payments?" (F)*
Avoiding Disaster	E K	2	Security will prevent a disaster.	*Yes, if [a disgruntled employee breech] happens once in five years, but it sets you back 10 years each time so [customers pay to prevent] it. (E)*

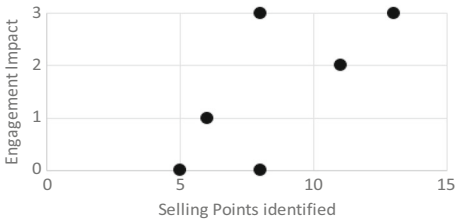

Fig. 7 Indicative Plot of Engagement Impact and Selling Points

management engagement?), the outcomes discussed in Table 5 in Section 5.3 suggest that this consideration was indeed effective.

Figure 7 plots this product management engagement impact against the number of selling points identified in each set of workshops[8]. As shown, those identifying more selling points tended to involve more product management engagement.

This does not provide evidence that the product management engagement impact was *caused* by the Security Promotion workshop. It is, however, reasonable to conclude that the Security Promotion workshop assisted in doing so. We conclude, therefore, that the answer to RQ 1.4 is yes, having developers consider the positive benefits of security and privacy mitigations can indeed lead to improvements in the security decision making process.

5.7 Threat assessment with developers

Considering the second Design Theory research question RQ 1.5 (*Can teams of developers produce threat assessments, risk-impact assessments, and benefit analyses with minimal guidance?*), we found that, surprisingly, all the sets of participants found effective ways to produce threat and risk-impact assessments. Even D, who are producing proofs of concept and are not domain experts for their products, had little difficulty:

> *We've identified huge risks that they need to consider before they ever get anywhere near an actual working product. (D)*

Team E learned and took away the prioritization process:

> *We had a follow-on session afterwards where we took everything away, … and sat down and thought "what do we need to do next". (E)*

In Group F, the facilitator produced a table of risks and impacts based on their discussion. Group G had no problem with risk assessments since two group members were familiar with the likelihood of attacks on the websites they managed. Group H and Group I simply had their most expert two members identify the most likely threats by placing asterisks on the flipchart. Group J had the cybersecurity specialists do the assessment. Group K successfully used post-it notes for the risk assessment, with separate dot-voting to identify the most likely and the most impactful threats.

It seems reasonable to conclude that the developers in the groups had the necessary skills and insights required. Thus, the answer to RQ 1.5 is affirmative: teams of developers can indeed produce adequate threat assessments, risk-impact assessment, and benefit analyses with minimal guidance.

[8] Excluding Group D for which data was not available.

5.8 Blockers and motivators related to security promotion

From our coding of the transcriptions of all the recorded workshops and interviews to address RQ 1.6 (*What are the 'blockers' and 'motivators' affecting product management engagement and other stakeholders as revealed in the workshops?*), we identified 30 blocker and 26 motivator statements, involving a total of 3166 words. Though blockers and motivators are in a sense opposites, they do not 'pair up' with each motivator addressing a specific blocker (Weir et al. 2019).

So, in answer to RQ 1.6, Table 8 lists the categories of blocker, ordered by how many were identified in each category, with a description of each category and example quotations from workshops. Table 9 does the same for motivators. Ten of the 30 blockers relate to poor communication. For motivators there is more variation, with 19 of the 26 split almost equally between friendly customers, policies, principled insistence, and value.

6 Answer to the primary research question

Returning to research question RQ 1 (*How can an intervention based on short workshops assist developers in identifying security issues, assessing them, and engaging product managers with those issues?*), we can now summarize the answer as follows.

Table 8 Blockers

Name	N	Description	Example Quotes
Communication	10	Difficulties in conveying security concepts or getting the right communication to achieve effective decisions	*[It] is difficult [to identify security requirements] as it requires a lot of conversation (I)* *The security thing is a bit of a taboo subject. (H)*
Multiple stakeholders	6	Different stakeholders may have different security appetites or needs; coordinating them is hard	*For some clients it's a really easy sell… But there's other clients: "Do I want to spend marketing budget on this?" (G)*
Freeloaders	4	Stakeholders expecting 'security for free'	*Some of our clients are now saying "You need to provide all this … for nothing because it's part of the security standard …" (G)*
Unknown cost/impact	4	Development teams may not have the ability to estimate costs; or may have inaccurate information about the likelihood of threats	*The mistake that customers have made with this app was to assume a small pilot study; then issue it to a big bunch of people… (E) We don't … give stakeholders an estimation of delivery time. We just chew through [work]. (J)*
PM time	3	product managers may not be available or have insufficient time to devote to the topic	*Some of us don't have access to a product owner. € [Customers] keep saying "We haven't had a chance to review what you sent us…" (D)*
Denial	2	Stakeholders refusing to accept a clear need for security	*No-one really cares about security until someone leaves their data on a train, anyway. (H)*
Practical	1	Practical issues, such as technology export restrictions	*If we want to put encryption into the firmware things, we need an export license. (H)*

Table 9 Motivators

Name	N	Description	Example Quotes
Friendly Customers	5	Focusing on customers who value security when it is explained to them	*Some clients are really good, and they will listen to best practice, and as soon as you start saying this … "Okay, right that's fine, tick, happy". (G)*
Policies	5	Externally enforced requirements for security	*If they've been in an organization with a PCI audit… they'll go to long, long lengths to avoid that. (G)*
Principled Insistence	5	Politely insisting on the need for the implementation of specific security features, on principle	*I think [customers and product managers] appreciate me saying "I don't think this is the best practice… You need to spend more money and do it this way". If I can back that up with the reasoning behind it, that is fine. (G)*
Value	4	Collaboratively identifying value for the stakeholders	*Things like single sign-on come to mind… We're improving the security. [It] actually makes life easier. (I)*
Communication	3	Improving communication: using handover documents, identifying security scope, and discussing consequences of poor security	*[For example] the handover document says, "The first thing you need to do is find a different way to send this", because … whoever develops this further needs to find a more secure way. (D)*
Logging Decisions	2	Keeping a log of security-related decisions, to support discussions and evaluation in future	*As long as you have made the decision based on the information … you have a reasoning behind why this is in, or why this isn't in. (E)*
Structured Workshops	2	Using facilitated workshops with stakeholders to inspire thinking on security issues	*I'm going to say to [my customer] "We're gonna have a security workshop. Come on have some lunch, bring [your developers] and we'll have a [workshop]". We won't charge … but at the end of it I'll bet [they'll] spend 20 grand because of the kind of client [they are] (G)*

Such an intervention is likely to need to address the design requirements from Section 3, including working with inexpert teams, being brief, and not requiring security experts or product managers. It should help teams to: understand security as a business driver, identify and prioritize types of security issues, cost solutions, and discuss those solutions effectively with product managers.

One possible implementation, as described in this paper, uses a game to promote understanding, and then short Threat Assessment and Security Promotion workshops. These workshops guide developers through identifying and prioritizing security issues for their own projects, costing solutions, and finding ways to promote security with product managers (Section 3).

Practical trials with teams in eight organizations have proved this implementation effective in improving product management engagement (Section 5.3). Participants required little explicit teaching to carry out the workshops (Section 5.7). They identified 8 categories of selling points for security (Section 5.5). Moreover, despite there being many blockers discouraging security improvement they also identified a similar number of motivators to encourage security improvement in future (Section 5.8).

Comparisons between different groups (Section 5.4) show that the workshops have greatest impact with groups with limited security expertise. Also, having the development team

managers as facilitators can be particularly effective in improving both product management engagement and threat assessment.

7 Discussion

7.1 Research method

As Section 4 explains, Design-Based Research (DBR) has been used mostly in the field of education research. While an intervention to change the behavior of software development teams is certainly a form of education, we are not aware of other researchers using DBR in this field.

In this research, as Section 5 shows, DBR has provided an effective basis for trialing, evaluating, and deducing theory from the use of an intervention. The discussion in that section showed that both Design Practice questions (RQ 1.1 through RQ 1.3) and Design Theory questions (RQ 1.4 through RQ 1.6) are of value, and contribute to our overall understanding.

7.2 Trustworthiness criteria and limitations

Table 10 explores five quality criteria for qualitative research of this kind (Denzin and Lincoln 2011; Stenfors et al. 2020) and highlights ways in which this paper satisfies those criteria. We can, however, identify three limitations in our deductions from the analysis:

- We have no way of evaluating either the completeness or accuracy of the threat assessment results. We believe that the developers' assessments were sufficient for the purpose of informing security improvements; that the consequences of getting a risk assessment wrong are much less than the consequence of not doing it at all; and that since product managers did engage well with the results (Section 5.6) the assessments were successful. However, this remains an outstanding question for future research.

Table 10 Quality Criteria

Criteria	What it means	Addressed in the Paper
Credibility	The research findings are plausible and trustworthy	Basis in extensive previous work (Weir et al. 2021a); explicit focus and answers to multiple research questions (Sections 4.2, 5.3–5.8); detailed and documented analysis (Sections 4.3, 4.4)
Dependability	The extent to which the research could be replicated in similar conditions	Workshop materials publicly available with full instructions (Section 6.4); analysis explained in detail with examples (Sections 4.3, 4.4)
Confirmability	There is a clear link or relationship between the data and the findings	Clear outcome summary (Fig. 6 and Table 5); use of quotes to substantiate results (Tables 5, 7, 8, and 9, Section 5.7)
Transferability	Findings may be transferred to another setting, context or group	Effectiveness in a wide range of situations (Section 5.1); analysis of where this is likely to be effective (Section 5.4)
Reflexivity	A continual process of engaging with and articulating the place of the researcher and the context of the research	Explicit descriptions of the researcher roles in the research (Sections 3.7, 4)

- Whilst in most cases product managers did engage with security in the development process (Section 5.6), we have no indication whether the resulting engagement led to more appropriate security in the resulting products. It is logical to assume that it would; but this research provides no evidence to support that assumption.
- We note also while we took care to distinguish security improvements caused by the interventions from other improvements (Section 4.3), in practice this distinction could not be exact. We also note the self-reported nature of the enhancements (Section 4.3).

The findings of this paper, therefore, form an existence proof: yes, the intervention can improve product management engagement. In addition, the range of different types of development involved in the trials prove there is a wide range of situations in which this intervention can work. We believe that the results we have found here justify further improvements of the intervention and its use in further development teams.

7.3 Practical value

Since our approach to the research is pragmatic, it is important to assess the practical value of these findings. We can identify three aspects that can be useful to professional developers, as follows:

1. The validation of the workshop package justifies its use in further software development teams;
2. The categorization of selling points (Table 7) potentially provides a basis for a structured approach for developers to assess selling points for security enhancements; and
3. The discussion of blockers and motivators (Tables 8 and 9) offers a practical simplification of a complex subject; the motivators table in particular offers practical ideas to allow a team to address security issues.

7.4 Further work

The package used in these trials has a practical limitation: it requires time input to train the facilitators, which potentially restricts its scalability to a wider audience of development teams. However, the workshops are peer-to-peer exercises where the facilitator only provides instructions rather than knowledge (Section 3.2). This offers the possibility of a version of the intervention that needs no direct training and therefore can scale without limit.

The authors have now created such a version with funding from the UK CyberASAP scheme; it is available online as the Developer Security Essentials package.[9] The full workshop package received an average of 15 downloads per month in 2021. In addition, the authors provide regular online facilitator training. As of the end of 2021, they had trained a total of 12 further facilitators; and two large multinational software development companies are deploying the package with their own teams.

The need to have researchers interview team members both before and after the interventions similarly limits the possible measurement of the success of such a new scaled-up intervention. An online, questionnaire-based version of the interviews can trade the flexibility

[9] https://www.securedevelopment.org/workshops/

of face-to-face interviews for the benefit of a large sample of results. Such a questionnaire has been implemented[10] and is free to use.

As discussed in Section 3.4, the Threat Assessment workshop uses only existing knowledge from the participants. This means that participants may fail to identify possible security issues, or wrongly assess the probability or impact of issues they do identify. This is particularly a problem with small companies, where there may be no security expertise available. To address this, participants would want evidence-based domain-specific knowledge of security issues and risk information. This would also require domain-specific nomenclature and definitions of security and privacy as used by developers and product managers. Current research by the lead author approaches these problems for a specific domain, Health IoT.[11]

8 Conclusions

This paper describes the outcomes from a project in which we, the authors, specified requirements, and designed a series of three workshops: a game to establish the importance and nature of security decisions; a Threat Assessment workshop to ideate and evaluate security risks in a specific project; and a Security Promotion workshop to find ways to discuss solutions with product managers (Section 3*)*. Using the Design-Based Research method (Section 4), we trialed the workshops in eight organizations, involving 88 developers.

The direct, Design Practice, outcomes of the trials were as follows:

- Five of the eight groups notably improved their threat assessment activities as a result of the interventions; six improved product management engagement (Section 5.3);
- Participants identified 50 different selling points, in 8 categories, of which the most prolific was 'Security Consultancy', improving customer relationships by impressing them with security expertise (Section 5.5); and
- Less security-expert groups appeared to benefit most from the workshops, and sessions appeared most effective when facilitated by team managers (Section 5.4).

The Design Theory findings from the research—to support further research and intervention development—included:

- Having developers identify selling points can indeed lead to improvements in product management engagement (Section 5.6);
- Teams of developers can produce threat assessments, risk-impact assessment, and benefit analyses with minimal guidance (Section 5.7); and
- A range of blockers, particularly problems with communication, challenge the introduction of security; however, there is a wide range, and similar numbers, of motivators to encourage it (Section 5.8).

We conclude that the intervention can be effective both in improving the security practice of development teams and in improving communication with product managers (Section 5.9).

[10] https://www.securedevelopment.org/security-assessment/
[11] https://lancaster.ac.uk/hipster

The findings from the project promise the possibility of a lightweight activity, that can easily be carried out by any development team, to help that team align their development security goals with their organization's business goals. One such implementation is now supported and freely available (Section 6.4), and this and similar interventions can help improve the security of the software on which we all rely.

Appendix 1: Interview Questions

Entry Interview

Introduction – establish context

- What is your current role, and what do you find yourself doing day-to-day? What's your involvement with this project?

Exploration

- Have you considered security for this project yourself? What's been done so far?
- In what ways do you consider security important for this product?

Experience

- What's the last time you came across a security issue in a project? Can you describe the issue?
- How did you deal with that issue?
- How confident are you about that solution?

Vision

- Let's imagine the project's finished, and it's been an excellent piece of work. What do you feel you'll have done related to security and privacy to get it that way?

Clarification (as appropriate)

- Oh, I see. Could you give an example?

Exit Interview

Introduction – establish context

- Now that we've been working together for a while, this is a discussion to see how things have progressed in the project.

Exploration

- What do you think has changed?

- What are your feelings about the change in the project?
- What did you make of the three activities we did: game, workshop, follow-ups?
- In what way might you have a better story on security now?

Experience

- What changes did you make as a result of the workshops and discussion?
- What exactly did you do?
- How did you go about implementing the changes?
- Why you chose to do those things?
- What is it that's better now as a result?
- Would you do something similar again?
- What would you do differently?
- How does this relate to these specific threats you've identified (from the threat modeling workshop)?

Vision

Let's imagine there's a team starting a similar project now, and you're advising the team coming in to help them improve their security. What would you recommend that's the same as we did, and how would you recommend improving it?

Acknowledgements We thank all the teams of developers and companies who contributed to this research. We also thank the editors and reviewers who helped us present the work effectively in this paper.

This work has been supported by the PETRAS National Centre of Excellence for IoT Systems Cybersecurity, which has been funded by the UK EPSRC under grant number EP/S035362/1.

Authors' contributions N/A

Funding (information that explains whether and by whom the research was supported)

This work has been supported by the PETRAS National Centre of Excellence for IoT Systems Cybersecurity, which has been funded by the UK EPSRC under grant number EP/S035362/1.

Data Availability All transcriptions and analysis were commercially confidential, several subject to Non-Disclosure Agreements.

Code availability Workshop materials are available at https://securedevelopment.org/workshops

Declarations

Conflicts of interest/competing interests None.

Ethics approval The project was approved by the Lancaster University Faculty of Science and Technology Research Ethics committee.

Consent to participate N/A

Consent for publication N/A

Bibliography

Ambreen T, Ikram N, Usman M, Niazi M (2018) Empirical research in requirements engineering: trends and opportunities. Requir Eng 23:63–95. https://doi.org/10.1007/s00766-016-0258-2

Ashenden D, Lawrence D (2013) Can we sell security like soap? A new approach to behaviour change. New Secur Paradigms Work 2013:87–94. https://doi.org/10.1145/2535813.2535823

Assal H, Chiasson S (2019) Think secure from the beginning: a survey with software developers. In: conference on human factors in computing systems (CHI). ACM. https://doi.org/10.1145/3290605.3300519

Bakker A (2018) Design research in education: a practical guide for early career researchers. Routledge, Abingdon

Barab S, Squire K (2004) Design-based research: putting a stake in the ground. J learn Sci 13(1):1–14. https://doi.org/10.1207/s15327809jls1301_1

Barbacci MR, Ellison R, Weinstock CB, Wood WG (2000) Quality attribute workshop participants handbook

Beck K, Fowler M (2001) Planning extreme programming. Addison-Wesley Professional

Becker I, Parkin S, Sasse MA (2017) Finding security champions in blends of Organisational culture. In: European workshop on usable security – EuroUSEC. https://doi.org/10.14722/eurousec.2017.23007

Beecham S, Baddoo N, Hall T (2008) Motivation in software engineering: a systematic literature review. Inf Softw Technol 50:860–878. https://doi.org/10.1016/j.infsof.2007.09.004

Bell L, Brunton-Spall M, Smith R, Bird J (2017) Agile application security: enabling security in a continuous delivery pipeline. O'Reilly, Sebastopol, CA

Beyer M, Ahmed S, Doerlemann K, Arnell S, Parkin S, Sasse A, Passingham N (2015) Awareness is only the first step: a framework for progressive engagement of staff in cyber security. Business white paper: Hewlett Packard

Brown AL (1992) Design experiments: theoretical and methodological challenges in creating complex interventions in classroom settings. J Learn Sci 2:141–178. https://doi.org/10.1207/s15327809jls0202_2

Bukhsh FA, Bukhsh ZA, Daneva M (2020) A systematic literature review on requirement prioritization techniques and their empirical evaluation. Comput Stand Interfaces 69:103389. https://doi.org/10.1016/j.csi.2019.103389

Caputo DD, Pfleeger SL, Sasse MA, Ammann P, Offutt J, Deng L (2016) Barriers to usable security? Three organizational case studies. IEEE Secur Priv 14:22–32. https://doi.org/10.1109/MSP.2016.95

Clarke V, Braun V, Hayfield N (2015) Thematic analysis. In: Smith JA (ed) qualitative psychology: a practical guide to research methods. SAGE publications, pp 222–248

Collins A (1992) Toward a design science of education. In: New Directions in Educational Technology. Springer, pp 15–22. https://files.eric.ed.gov/fulltext/ED326179.pdf

Conradi R, Dybå T (2001) An empirical study on the utility of formal routines to transfer knowledge and experience. ACM SIGSOFT Softw Eng Notes 26:268–276. https://doi.org/10.1145/503271.503246

Dabbagh M, Lee SP, Parizi RM (2016) Functional and non-functional requirements prioritization: empirical evaluation of IPA, AHP-based, and HAM-based approaches. Soft Comput 20:4497–4520. https://doi.org/10.1007/s00500-015-1760-z

Davison RM, Martinsons MG, Kock N (2004) Principles of canonical action research. Inf Syst J 14:65–86. https://doi.org/10.1111/j.1365-2575.2004.00162.x

De Win B, Scandariato R, Buyens K, Grégoire J, Joosen W (2009) On the secure software development process: CLASP, SDL and touchpoints compared. Inf Softw Technol 51:1152–1171. https://doi.org/10.1016/j.infsof.2008.01.010

Denzin N, Lincoln Y (2011) The Sage handbook of qualitative research

Design-Based Research Collective (2003) Design-based research: an emerging paradigm for educational inquiry. Educ Res 32(1):5–8. https://doi.org/10.3102/0013189X032001005

Dybå T (2005) An empirical investigation of the key factors for success in software process improvement. IEEE Trans Softw Eng 31:410–424. https://doi.org/10.1109/TSE.2005.53

Easterbrook S, Singer J, Storey M-A, Damian D (2008) Selecting empirical methods for software engineering research. In: Guide to advanced empirical software engineering. Springer, London, pp 285–311. https://doi.org/10.1007/978-1-84800-044-5_11

Ejersbo LR, Engelhardt R, Frølunde L, Hanghøj T, Magnussen R, Misfeldt M (2008) Balancing product design and theoretical insights. In: The Handbook of Design Research Methods in Education. Routledge, pp. 149–164

Fisher R, Ury WL, Patton B (2011) Getting to yes: negotiating agreement without giving in. Penguin

Fogg BJ (2009) A behavior model for Persuasive design. In: international conference on Persuasive technology - Persuasive. ACM, pp 40:1–7. https://doi.org/10.1145/1541948.1541999
Franqueira VNL, Tunnicliffe P (2015) To Flip or not to Flip: a critical interpretive synthesis of flipped teaching. In: Smart Education and Smart e-Learning. Springer, pp. 57–67. https://doi.org/10.1007/978-3-319-19875-0_6
Frey S, Rashid A, Anthonysamy P, Pinto-Albuquerque M, Naqvi SA (2017) The good, the bad and the ugly: a study of security decisions in a cyber-physical systems game. IEEE Trans Softw Eng 45(5):521–536. https://doi.org/10.1109/TSE.2017.2782813
Gwet KL (2014) Handbook of inter-rater reliability: the definitive guide to measuring the extent of agreement among raters. Advanced Analytics LLC
Haines S (2014) The product Manager's desk reference, Second ed. McGraw-Hill, New York
Hall T, Sharp H, Beecham S, Baddoo N, Robinson H (2008) What do we know about developer motivation? IEEE Softw 25:92–94. https://doi.org/10.1109/MS.2008.105
Herzberg F (2017) Motivation to work. Routledge
Hubbard DW, Seiersen R (2016) How to measure anything in cybersecurity risk. John Wiley & Sons
ISO/IEC (2008) 21827:2008 - Systems Security Engineering - Capability Maturity Model
Kelly AE, Lesh RA, Baek JY (2008) Handbook of design research methods in education: innovations in science, technology, engineering, and mathematics learning and teaching. Routledge
Kirlappos I, Beautement A, Sasse MA (2013) "Comply or die" is dead: long live security-aware principal agents. In: Financial cryptography and data security. Springer, Berlin, Heidelberg, pp 70–82. https://doi.org/10.1007/978-3-642-41320-9_5
Kluyver T, Ragan-kelley B, Pérez F et al (2016) Jupyter notebooks: a publishing format for reproducible computational workflows. In: Positioning and power in academic publishing: players. IOS Press, Agents and Agendas, pp 87–90
Lopez T, Sharp H, Tun T, Bandara A, Levine M, Nuseibeh B (2019a) Hopefully we are mostly secure: views on secure code in professional practice. In: Workshop on Cooperative and Human Aspects of Software Engineering - CHASE. IEEE, pp. 61–68 https://doi.org/10.1109/CHASE.2019.00023
Lopez T, Sharp H, Tun T et al (2019b) Talking about security with professional developers. In: Workshop on Conducting Empirical Studies in Industry - CESSER-IP. IEEE Computer Society, Montreal, QC, Canada
McSweeney B (1999) Security, identity, and interests: a sociology of international relations. Cambridge University Press https://doi.org/10.1109/CESSER-IP.2019.00014
Mead NR, Stehney T (2005) Security quality requirements engineering (SQUARE) methodology. In: SESS 2005 - proceedings of the 2005 workshop on software engineering for secure systems - building trustworthy applications. Pp 1–7. https://doi.org/10.1145/1082983.1083214
Mellado D, Fernández-Medina E, Piattini M (2006) Applying a security requirements engineering process. In: lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics). Pp 192–206 https://doi.org/10.1007/11863908_13
Microsoft (2018) Microsoft security intelligence report, Volume 23. https://info.microsoft.com/rs/157-gqe-382/images/en-us_cntnt-ebook-sir-volume-23_march2018.pdf. Accessed 6 Mar 2019
Nhlabatsi A, Nuseibeh B, Yu Y (2012) Security requirements engineering for evolving software systems: a survey. In: Security-Aware Systems Applications and Software Development Methods. IGI Global, pp. 108–128. https://doi.org/10.4018/978-1-4666-1580-9.ch007
Oxford Languages (2011) Concise Oxford English Dictionary
Pfleeger SL, Sasse MA, Furnham A (2014) From weakest link to security Hero: transforming staff security behavior. J Homel Secur Emerg Manag 11:489–510. https://doi.org/10.1515/jhsem-2014-0035
Poller A, Kocksch L, Türpe S, Epp FA, Kinder-Kurlanda K (2017) Can security become a routine? A study of organizational change in an agile software development group. In: Conference on computer supported cooperative work - CSCW. ACM, Portland Oregon USA, pp 2489–2503. https://doi.org/10.1145/2998181.2998191
Rauf I, Petre M, Tun T et al (2022) The case for adaptive security interventions. ACM Trans Softw Eng Methodol 31:1–52. https://doi.org/10.1145/3471930
RiskBased Security (2020) 2020 Mid Year Data Breach Report
Shostack A (2014) Threat modeling: designing for security. John Wiley & Sons
Shreeve B, Hallett J, Edwards M, et al (2020) The best laid plans or lack Thereof: Security Decision-Making of Different Stakeholder Groups. IEEE Trans Softw Eng. https://doi.org/10.1109/TSE.2020.3023735
Springer O, Miler J (2018) The role of a software product manager in various business environments. In: proceedings of the 2018 federated conference on computer science and information systems, FedCSIS 2018. Polish information processing society, pp 985–994
Stack Overflow (2016) Annual Developer Survey. https://insights.stackoverflow.com/survey/2016. Accessed 17 Jun 2020
Stenfors T, Kajamaa A, Bennett D (2020) How to … assess the quality of qualitative research. Clin Teach 17:596–599. https://doi.org/10.1111/TCT.13242
Such JM, Gouglidis A, Knowles W et al (2016) Information assurance techniques: perceived cost effectiveness. Comput Secur 60:117–133. https://doi.org/10.1016/j.cose.2016.03.009

Tietjen MA, Myers RM (1998) Motivation and job satisfaction. Manag Decis 36:226–231. https://doi.org/10.1108/00251749810211027

Türpe S, Kocksch L, Poller A (2016) Penetration tests a turning point in security practices? Organizational challenges and implications in a software development team. In: Workshop on Security Information Workers - SIW. USENIX Association

van der Linden D, Anthonysamy P, Nuseibeh B, et al (2020) Schrödinger's security: opening the box on app developers' security rationale. In: International Conference on Software Engineering - ICSE. IEEE

Veracode (2018) State of Software Security Report Volume 9. https://info.veracode.com/report-state-of-software-security-volume-9.html. Accessed 6 Feb 2019

Viera AJ, Garrett JM (2005) Understanding Interobserver agreement: the kappa statistic. Fam Med 37(5):360–363

Wang F, Hannafin MJ (2005) Design-based research and technology-enhanced learning environments. Educ Technol Res Dev 53:5–23. https://doi.org/10.1007/BF02504682

Weir C, Becker I, Blair L (2021a) A passion for security: intervening to help software developers. In: 2021 IEEE/ACM 43rd international conference on software engineering: software engineering in practice (ICSE-SEIP). IEEE, pp 21–30. : https://doi.org/10.1109/ICSE-SEIP52600.2021.00011

Weir C, Becker I, Noble J, et al (2019) Interventions for long-term software security: creating a lightweight program of assurance techniques for developers. Softw - Pract Exp 275–298. : https://doi.org/10.1002/spe.2774

Weir C, Hermann B, Fahl S (2020a) From needs to actions to secure apps? The effect of requirements and developer practices on app security. In: 29th USENIX security symposium (USENIX security 20)

Weir C, Knight J, Ford N (2021b) Developer Security Essentials. https://www.securedevelopment.org/workshops/. Accessed 9 Jun 2021

Weir C, Noble J, Rashid A (2020b) Challenging software developers: dialectic as a Foundation for Security Assurance Techniques. J Cybersecurity 30. https://doi.org/10.1093/cybsec/tyaa007

Xie J, Lipford HR, Chu B (2011) Why do programmers make security errors? In: IEEE symposium on visual languages and human centric computing. Pittsburg, PA, USA, pp. 161–164. : https://doi.org/10.1109/VLHCC.2011.6070393

Yskout K, Scandariato R, Joosen W (2015) Do security patterns really help designers? In: International conference on software engineering - ICSE. IEEE, Firenze, Italy, pp 292–302. https://doi.org/10.1109/ICSE.2015.49

Charles Weir has researched Developer Centered Security at Lancaster University, UK, since 2015. Prior to that he had 30 years in industry as a researcher, software architect, design consultant and company CEO, specializing in software development, especially for terminals and mobile devices. He was technical lead for the first smartphone, led the development of the first mobile money app for Android, and ran a successful software development company averaging 20 employees for 17 years.

Ingolf Becker is a Lecturer in Security and Crime Science at University College London, UK. He has been studying the interactions between security and business processes in organizations since 2013. This work has led him to collaborate with critical national infrastructure companies to technology multinationals and SMEs. Throughout his work qualitative research methodologies feature heavily, allowing him to understand the motivations, capabilities and limitations of individuals that are key to effective security decision making.

Lynne Blair is a Senior Lecturer at Lancaster University, UK. She specializes in software education, and co-leads Lancaster's involvement in the Institute of Coding, with a focus on widening participation. Much of her work is on human aspects of computing such as personal and social implications of our digital economy on community values and integrity, wellbeing, and environmental implications regarding sustainability in digital innovations.