Usability and Security by Design: A Case Study in Research and Development

Shamal Faily
Bournemouth University
sfaily@bournemouth.ac.uk

John Lyle, Ivan Fléchais, Andrew Simpson
University of Oxford
firstname.lastname@cs.ox.ac.uk

***Abstract*—There is ongoing interest in utilising user experiences associated with security and privacy to better inform system design and development. However, there are few studies demonstrating how, together, security and usability design techniques can help in the design of secure systems; such studies provide practical examples and lessons learned that practitioners and researchers can use to inform best practice, and underpin future research. This paper describes a three-year study where security and usability techniques were used in a research and development project to develop *webinos* — a secure, cross-platform software environment for web applications. Because they value innovation over both security and usability, research and development projects are a particularly difficult context of study. We describe the difficulties faced in applying these security and usability techniques, the approaches taken to overcome them, and lessons that can be learned by others trying to build usability and security into software systems.**

***Keywords*—Security; Usability; Context of Use; Personas**

I. Introduction

The security community has long recognised the need for incorporating a human dimension into its work. Surprisingly, however, while understanding how to design for user experience has been recognised as an important area for study, there has been comparatively little work examining the practical activities associated with building usable and secure systems. This observation was highlighted by Birge [2], who noted that there has been considerable research in Human-Computer Interaction and Security (HCISec) in recent years, but the bulk of this work has pertained to studying the usability of security controls and conceptual investigations about terms such as 'trust' and 'privacy', rather than activities associated with designing systems. In particular, much of this research focuses on the needs of end-users, rather than those of *designers* — few studies have attempted to tackle the question of how the latter should approach both usability and security as design concerns. At a NIST-hosted event on aligning software and usable security with engineering in 2011 [15], several recommendations for progressing the state-of-the-art were proposed, with one being that more case studies demonstrating the efficacy of best practice should be published. By showing how techniques were (or were not) applied successfully in contemporary design situations, practitioners could be better informed when selecting approaches for their own projects, while researchers could better understand the problems faced and develop research agendas for dealing with them.

An interesting source of case studies comes from collaborative research and development (R&D) projects with teams drawn from both industry and academia. The importance of security has been recognised in recent years, partly due to a number of high-profile incidents reported in the media. Exploring the security and privacy implications of planned innovations is now a common pre-requisite for the funding of R&D projects. Unfortunately, designing a project for security and usability is easier said than done. In many conventional projects, security and usability are not considered primary goals, making them likely candidates for sacrifice in the rush to meet project deadlines. Unlike most conventional projects, however, R&D projects usually tackle risky, ill-defined problems with the aim of achieving the greatest possible impact. But impact can mean different things to different people: industrial partners measure success by developing technology which promises to be commercially successful, whereas academic partners might use the technology to explore research questions that surface during development. This apparent conflict can mean that, when facing impending deadlines, different partners have different priorities and areas of interest. Because this can further undermine quality concerns such as security, usability, or performance, creative solutions to design problems need to be explored to account for potential conflict.

This paper describes the challenges faced designing security and usability into *webinos*: a software environment for running web applications securely across different device platforms. We begin in Section II by reviewing the issues faced in projects where the primary aims are to innovate, as well as the experiences of the HCISec community in contending with both security and usability during such projects. In Section III, we then present the *webinos* project and its objectives, the design approach taken to develop the *webinos* platform, and the part played by security and usability design techniques. We describe the difficulties faced in applying these techniques in Section IV, together with the approaches taken to tackle them. Finally, in Section V, we reflect on lessons for those looking to align security and usability design in projects of this nature.

USEC '15, 8 February 2015, San Diego, CA, USA
Copyright 2015 Internet Society, ISBN 1-891562-40-1
http://dx.doi.org/10.14722/usec.2015.23012

II. Related work

A. Innovation design in EU research projects

Designing for innovation might appear to be synonymous with general information system design, but several differences have been identified between the roles played by system *architects* and *entrepreneurs* [9] in conventional and innovation design respectively. The most prominent of these differences is that architects are system-centered and their architectures are realisations of system goals, whereas entrepreneurs are opportunity-centered, such that their architectures fit into an innovation strategy. This distinction is important: while architectural design is concerned with shaping an architecture to fit a socio-technical environment, innovation is equally concerned with shaping the socio-technical environment around the architecture. Because of this difference in perspectives, designing for innovation is often hampered by what has been described as the *innovation design dilemma* [14]: structured processes (such as those typically associated with secure system design) generate few ideas, while more unstructured processes generate more diversity – but at the cost of conflict that may hamper the implementation of innovation.

The priority of innovation over system design qualities is evident when considering the role usability has played in several European security and privacy projects. The PrimeLife project [22] demonstrated how privacy technologies can enable people to control their on-line personal information based on their legal rights. Based on their experience developing a number of Privacy Enhancing Technology (PET) exemplars, the PrimeLife team identified several useful heuristics and idioms when designing and evaluating user interfaces for PETs. Usability researchers played an important role in evaluating PrimeLife, but there is less evidence of their influence in the design of the exemplars themselves. Representations of users were created during the project to guide design decisions. However, in an example of privacy in social software [3], one of these representations is used only to illustrate access control policies in web forums, rather than describing how it influenced design decisions. While it is possible that these representations were used to inform architectural and application design, this is not reported.

B. Security and Usability Design

While the need for usable security had long been recognised, the work of Zurko and Simon on user-centered security — security models, mechanisms, systems and software having usability as a primary motivation or goal — was one of the first acknowledgements of the part designers can play in achieving this [37]. As well as reinforcing the need for security models that are sensitive to the mental models of different types of users, they propose combining security design techniques with established design techniques from HCI. Work by Sasse *et al.* [25] has illustrated how such usability design approaches can be applied to the design of security mechanisms. Based on the empirical findings from several studies about password usage, they found that framing the design of security controls from technology, user, goal, task, and context perspectives can lead to useful insights. Considering security as an enabling task that needs to be completed before a main task begins explains why users choose to circumvent controls getting in the way of their work.

Although the work of Sasse *et al.* and Zurko and Simon continues to inspire HCISec research, much of this community's work focuses on studying the usability of security controls and interfaces rather than activities associated with designing interactive secure systems. Birge [2], in outlining five broad categories of research in HCISec, observed that the only one that mentions *design* is the 'Usability and Design Studies' category: exploring how traditional usability methods can be used to make decisions about user interfaces with security or privacy implications. In light of the lack of published work illustrating how usable and secure systems might be designed, a workshop was held in 2011 to explore whether it was possible to blend security, usability and software engineering lifecycles [15], with a number of informal anecdotes about promoting usability design during secure software development being discussed. Case studies exploring such insights are valuable, but still in short supply.

C. Lessons learned from e-Science and NeuroGrid

Some problems faced by R&D projects have already been encountered by the e-Science community, which is concerned with building IT infrastructures that facilitate global scientific collaboration. e-Science projects need to address two challenges relating to usable security design. First, global collaboration entails building coalitions and working partnerships between stakeholders who are distributed and culturally diverse. Cultural differences can lead the same artifacts in the same apparent context of use being perceived and used in different ways based on the norms and values of different scientific communities [8]. Distributed stakeholders also means that carrying out formative evaluation activities to glean and reason about these differences can be expensive and time-consuming. Second, security is not usually treated as a priority when attempting to cast light on scientific uncertainties. Prioritising core functionality does not mean that security is ignored in e-Science, but, as is suggested in [17], there is a tendency to treat design for security in an ad-hoc manner. Given the different perceptions stakeholders might hold about assets in an e-Science project, security design decisions might be ill-suited to the assets needing protection. For example, some stakeholders may not believe highly aggregated data sources to be a valuable target for an attacker; however, with the right search criteria, they could be de-anonymised to the detriment of another stakeholder.

The NeuroGrid project is a useful exemplar of the implications of usability and security in the design of e-Science projects. This project is also the subject of one of the few published studies describing the application of security and usability practices in e-Science projects [30]. NeuroGrid was a three-year project funded by the UK Medical Research Council to develop a Grid-based collaborative research environment; this was designed to enhance collaboration both within and between clinical research communities [18]. The sensitivity and distributed nature of the clinical data drove the need to find secure and effective ways of accessing and managing it. The project was ultimately successful, but several problems stymied the adoption of NeuroGrid in its targeted research communities; these included failing to specifically address usable security problems.

Although usability researchers had elicited requirements from a wide range of project stakeholders, many security decisions with an implication on user interaction had been made before their engagement. As a result, the usability researchers felt they had little control over any design or implementation decisions affecting the security architecture; this meant there was little direct engagement with the team responsible for building and maintaining NeuroGrid's core security mechanisms. Because of this, when scientists expressed problems using the Public Key Infrastructure during formative evaluation activities, application developers were left with the responsibility of addressing them. While simple interface changes helped explain terms to non-specialists, as did writing documentation to explain security controls, developers were also left with the responsibility of implementing components to manage users' digital certificates on their behalf; this effectively replaced digital certificates with passwords as an authentication mechanism. Although this appeared to improve user perception of authentication usability, the design and implementation of these security components was not subject to the same security evaluation criteria as the rest of the security architecture, and, as such, the vulnerabilities associated with it were largely unknown.

III. Approach

In this section, we describe how security and usability design activities were incorporated into the development process of *webinos*. We begin by summarising the *webinos* project and the motivations behind it, and then explain the development process, before describing the security and usability design activities in more detail.

A. About webinos

The *webinos* project was funded by the EU with a project team drawn from 24 organisations across Europe, including universities, mobile network providers, handset and automotive manufacturers, mobile software houses, and market analysts. The project ran from September 2010 to August 2013. The primary objective was to develop a federated software platform for running web applications consistently and securely across mobile, PC, home media, and in-car systems. The *webinos* platform provides a software runtime environment that allows the discovery of devices and services based on technical and contextual information. The platform also offers a set of APIs providing access to cross-user, cross-service, and cross-device functionality. More information about the *webinos* architecture can be found at [35].

Because *webinos* applications can access physical and informational resources across different devices, and this information may describe a user's personal habits and preferences, managing access is a complex security and usability design problem. Questions are also raised about the needs of application developers who must request permissions for their applications to access user data and provide facilities for users to manage their security and privacy. This is exacerbated by the lack of prior art and experience in cross-platform and multi-device personal networking, especially given the different physical and social contexts of use.

The large number of collaborating partners, the innovative nature of the project, and the significant security concerns all influenced the need to integrate security and usability design techniques in the development process.

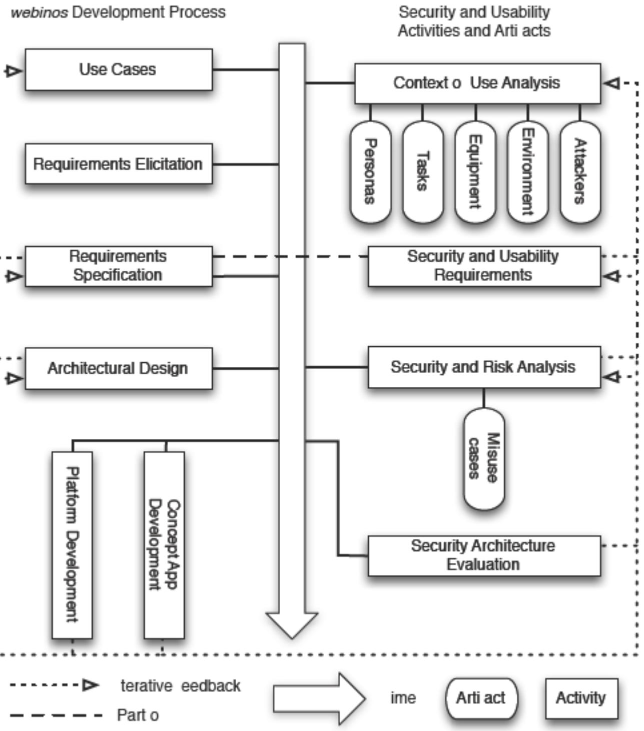

Fig. 1. *webinos* development process

1) The Design and Development process: A first version of the *webinos* platform was publicly released in March 2012, following an 18-month development effort that is summarised in Figure 1.

A use-case driven approach was used to elicit and specify software requirements for the initial version of *webinos*. During the first six months of development, over 80 use cases were drafted and ranked by importance and novelty by the project team. As the project scope became clearer, the number of use cases were reduced to 33. These final use cases, and the process used to create them is described in [33].

The use cases, along with an industry and software ecosystem analysis, informed an up-front requirements elicitation and specification activity for the *webinos* platform. In the subsequent six months, these requirements informed the development and architectural design of *webinos*, its APIs, and the security and privacy frameworks. In the last six months, two concurrent activities took place: platform development, and development of a collection of concept apps. Both the platform and concept apps were implemented via an agile model based on the Scrum method [27], with sprints of approximately two weeks. Scrum teams were created for representative *webinos* platforms, with the core architectural components and features developed being derived from the architectural design documentation. Insights from the implementation led to updates to the use cases and requirements, which were delivered at the same time as the platform. In parallel, based on the API specifications, several concept apps were developed to demonstrate *webinos*' capabilities and provide feedback to

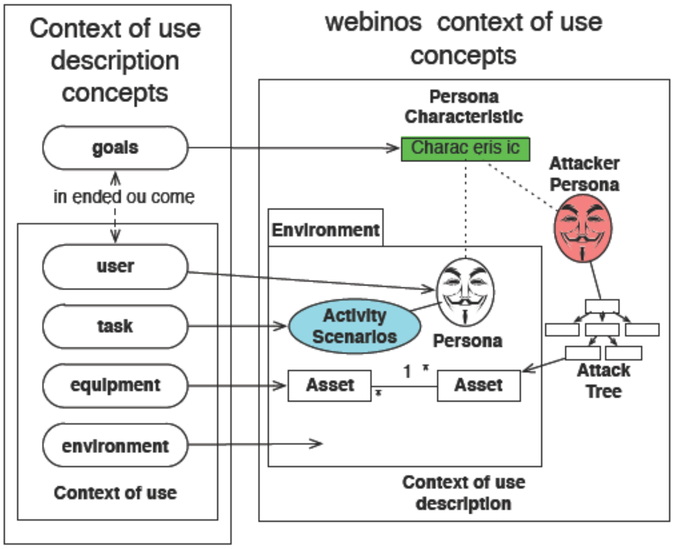

Fig. 2. Context of use description concepts

platform developers.

2) The Usable Security design team: Security and usability design activities were integrated into the design and development process. These activities took place during the project's first two years, and were carried out by a team of 10 practitioners and academics drawn from five different organisations; this team included some of the authors of this paper. Because the team was responsible for both security *and* usability, the team would avoid the engagement problems found on NeuroGrid.

The distributed nature of the team meant that on-going progress was discussed in fortnightly telephone conferences during the first year. Although all team members had an information security background and an appreciation of the importance of usability in the design process, only three had any practical experience of applying usability design techniques to a real project. For this reason, detailed guidelines and examples of how to use the techniques were documented on a project wiki, and time was set aside at each telephone conference for answering questions about them. In the second year, telephone conferences and meetings were less frequent, and multi-day meetings were held during the project meetings that took place throughout the year. In the remaining 18 months of the project, the *webinos* platform was continually refined, with its source code being released to the community as multiple open-source projects [34].

B. Context of Use Analysis

To begin driving the project's security and usability design activities, the team created an adapted *context of use* description. This description, which is specified in [32], was used to support the specification and architectural design of *webinos* by capturing and modelling the security expectations of *webinos* stakeholders.

Context of use descriptions describe the characteristics of a system's intended users, the tasks they are expected to perform, and the environments a system is expected to operate in [16]. Although context of use descriptions were devised to support formative and summative usability evaluation activities, the team wanted to use this description to motivate and justify security and usability design decisions; this would avoid issues found in previous European security and privacy projects where usability artifacts were disconnected from subsequent design decisions. Because of this, the context of use description developed for *webinos* was aligned with usability and secure system design techniques and concepts in a number of different ways, as summarised in Figure 2.

1) Users: In standard context of use descriptions, skills, knowledge, and personal attributes are captured about *primary* (direct) or *secondary* (indirect) users. To elicit and specify information about users, a two-step approach was taken. First, the project team considered the project scope and identified two main categories of users that *webinos* should be designed for (app developers and app end-users). Second, a collection of 12 *personas* were developed to elicit and specify skills and attributes of prospective users. Personas are specifications or archetypical users that embody their goals and needs [10]; these were selected as user representations because previous work indicates that developing and applying them can contribute positively to secure system design activities [7]. To model user goals succinctly, the team structured the data contributing to the characteristics of each persona; each characteristic corresponded to an activity, aptitude, attitude, motivation, or skill-related personal goal. This process is described in more detail in [10].

2) Tasks: Tasks include attributes such as name, breakdown, frequency of use, duration, and physical and mental demands. Although scenario-based task descriptions are analogous to use cases, insufficient contextual information was available to elicit task duration and demand information from the use case template used for *webinos*, which was based on [5]. Instead, a collection of 10 activity scenarios [24] was created; these were written based on the project team's domain expertise and conjectured how the previously developed personas might use the *webinos* platform. Using both the narrative and the characteristics of each participating persona, a table categorising scenario's duration, frequency of use, and physical and mental demands was completed for each scenario.

3) Environments: Environments are not formally defined by the context of use description, but are recognised as part of a work system consisting of users, equipment, tasks, physical and social environment, and the goals the work system needs to achieve [6]. Conceptually, work systems are analogous to contexts of use, so environments can be considered as the social or physical surroundings within which *webinos* is used. From a specification perspective, *webinos* assumes the presence of a single physical environment, but everyone on the project team viewed *webinos* differently depending on their individual or organisational stake. To better understand how *webinos*' surroundings influenced its design, the team carried out a domain modelling exercise of *webinos* to identify key concepts in five functional areas and five target application areas; functional areas included 'Development & Deployment' and 'Media Consumption & Creation', and target application areas included 'Navigation' and 'Home media and interactive TV'. The key concepts and relationships were drawn from the

Fig. 3. Posters of personas characteristics, goals, and expectations

scenarios and use cases, and modelled as UML class diagrams; these included hardware, software, or service concepts of importance which constrain or reference *webinos*.

4) Equipment: Equipment is concerned with the specification of hardware, software, and services associated with the system being designed. Specifying the context of use involves describing, albeit in broad terms, how related systems, such as applications and devices, interface with *webinos*. The domain models described in Section III-B3 formed the basis of UML class diagrams of architecturally significant assets. Each 'asset model' diagram was based on a particular environment of interest. Of the ten environments originally domain modelled, asset models were based on three functional areas, and three target application areas. Attributes in the class diagram were used to describe the security (Confidentiality, Integrity, Availability, Accountability) and privacy (Anonymity, Pseudonymity, Unlinkability, Unobservability) properties of each asset.

5) Attackers: The context of use description helps identify vulnerabilities arising from usability problems, but says little about how these might be exploited. The team believed viewing such vulnerabilities from the perspective of an attacker might provide additional insights into these problems, so a set of 6 *attacker* personas were incorporated into the context of use description. These were based on threat agents specified by the OWASP project [20], and grounded in publicly available open source intelligence. Attack trees [26] were created to visually and systematically represent how these personas might launch certain attacks. More information on how attacker personas were created and used can be found in [1].

C. Building Security into the Architectural Design of webinos

During the use case and requirement specification stage, *webinos* was treated as a black box, the tasks and personas within the context of use description served as a reminder of the expectations of prospective users and application developers. To remind the project team about these expectations, A4 posters were created for each persona. These summarised the key characteristics, goals, and expectations of each persona; as illustrated in Figure 3, project partners were encouraged to place these posters in prominent locations around their offices.

Informed by the use cases and context of use description elements, Security and usability requirements were elicited and specified along with the broader functional and non-functional requirements. Later, the architectural design considered *webinos* as a *white box*, which allowed the team to perform a detailed security analysis. New questions were raised about user expectations at this stage; answering these required the team to combine the security analysis with their previous usability work.

The security analysis began by running a two-day participatory risk analysis workshop; this drew on the team's domain and platform security expertise. The asset models developed as part of the context of use description were updated based on insights from the newly emerging *webinos* architecture. The previously created context-specific asset models were qualitatively rated based on their perceived security and privacy properties. Sensitised by this analysis, the team identified several new vulnerabilities, together with a number of attacks that the attacker personas might carry out. Risks were identified based on threats exploiting vulnerabilities in particular contexts of use. To explore their human impact, each of the 12 risks elicited were validated using a misuse case: a sequence of actions that an attacker can perform which will cause harm to stakeholders when the sequence is allowed to complete [29]. Although misuse cases are traditionally used to *elicit* risks, using this approach for risk *validation* not only places the risk in a human context, it also sanity-checks the analysis contributing to the definition of each risk. The results of this analysis were used to guide the design and evaluation of the security architecture. For example, one misuse case described an instance of fraud due to abuse of the *webinos* payment API. In exploring how this misuse case might be possible, several team members speculated that a native app developer might find grappling with the JavaScript API difficult, leading to concerns that security would be sacrificed if it steepened this persona's learning curve. Not only did this discussion help to cement the importance of usability in the security of *webinos*, but it also helped to reinterpret the *webinos* architecture and support further analysis into how vulnerable code might be developed. These risks and misuse cases are described in more detail in [31].

D. Supporting Platform and Application Development

Although security and usability design activities did not formally take place in the later stages of the project, the results from the previous stages were used to support the on-going platform and concept app development activities.

Several of the platform development decisions were motivated by the need to mitigate the previously defined misuse cases. One of these described a persona (*Justin*) habitually using a mobile broadband connection to install interesting applications on his smartphone as part of his morning commute. When installing a *webinos*-enabled diary application for the first time, *Justin* confused some of the dialog controls for setting up privacy preferences with those associated with exporting/importing data from one application to another. Consequently, as a result of habituation, *Justin* unintentionally accepted privacy-invasive default settings defined by this application's developer. This led to discussions within the development team about the application installation process and the granting of access permissions. Personas and activity scenarios were also useful for grounding developer discussion when the lack of visibility of dependent applications and users might have led to an inappropriate expansion of the *webinos* runtime environment's scope.

Concept app development was supported by directly applying the context of use description, particularly the personas, to motivate application design decisions. One of these apps was a travel game application called *Kids in Focus* where one player using an in-car telematics system could play card games with someone using a home media system. The game was specifically design to support the in-car systems of a mother (*Helen*) who wished to keep her young son entertained on a long car journey, by talking and playing a card game with his grandfather (*Peter*) who was at home. By encouraging the developers to design specifically for the personas and their contexts of use, it was possible to highlight which controls were either necessary or unnecessary. For example, this concept app was built using pre-existing components that assumed authentication would be necessary. However, application authentication would not be required because the *webinos* platform had already verified the player identities before commencing the game. As the mobile component of the application would be used by a young child, a number of privacy design considerations were identified during the application design, such as whether it was appropriate to switch game play to an 'auto-pilot' when connection to *Peter* was lost. Thinking of both the young son and the security and privacy asset values in the related environment helped motivate the design decisions that were eventually made. Similarly, defining the explicit elements of the context of use made it easier for developers to identify what information and resources would be required by the application, and whether or not these conflicted with the privacy preferences of the personas.

E. Releasing webinos Design Data

In addition to making the source code of *webinos* available to the community, its specification and architectural design have also been made available as public deliverables.[1] However, to encourage take-up of *webinos*, the fine-grained design data – upon which its architecture is based – was also released with the source code to potential *webinos* users and developers. In doing so, the intent was to equip the community with the same data and tools that the project used to design and build *webinos*. In addition to its software requirements, this design data included XML models of the context of use specifications, as well as the threat models used to evaluate the *webinos* architecture's security. This allows developers to carry out their own risk analysis on later versions of *webinos* specifications, to determine whether *webinos* meets their own security and privacy expectations, as well as those of the personas in the context of use description.

IV. Difficulties

While the approach taken was largely successful, it was not entirely problem-free. Indeed, we would have been surprised if the results had indicated otherwise. Because many of the challenges faced were of a project management nature, we have chosen to focus on four specific difficulties associated with 'designing in' security and usability.

A. User research is not easy

To build the models described in Section III-B, it was necessary to collect data about the people associated with *webinos*, and their day-to-day activities. This data would also be useful for security design because it reveals insights about possible vulnerabilities associated with device interactions in different contexts of use [11]. Despite this, the team faced a significant challenge conducting the research necessary to collect data directly from potential end-users. We believe there were three main reasons for this difficulty.

First, because the team members did not have the resources to undertake all the user research activities themselves, data collection activities needed to be distributed among the whole team. Unfortunately, not only was the team broadly unfamiliar with usability design techniques and methodologies, they also were not working on these design activities on a full-time basis.

Second, although some primary data was collected from prospective developers as part of an industry landscape analysis conducted by a different *webinos* team member, this data was collected via semi-structured interviews that were not designed for eliciting security concerns. The only opportunity to address this came at a late stage, and allowed the team to include two very specific questions about how developers design their applications for security and privacy, however there was no opportunity to follow-up on any interesting points. Moreover, because detailed transcripts were not recorded by the interviewers, only a very small number of security and privacy developer perceptions were elicited from the 21 interviews conducted.

Third, there was little guidance about how primary security data might be collected for the planned design activities. Existing work on collecting primary data for security analysis tends to focus on quantitative data from software and hardware sources, rather than qualitative data from people. Further, such data tends to be analysed to identify attackers, rather than to identify behavioural patterns. Work by the Hackers Profiling Project [4] describes a systematic approach for collecting primary data about hackers and using this to develop profiles; unfortunately, replicating this work would have been infeasible given time constraints.

To overcome the problem of limited primary data, the team tried to make the most of what data it had access to: both making use of additional data sources to inform and justify design decisions, and by reusing data it had collected for other purposes in *webinos*. Some of the additional sources of data used were interview and focus group reports from prior research that were relevant to representative users, email discussions between domain experts about archetypical users, and online resources about representative user communities. One of the main sources of reused data was the *webinos* design

[1] These are downloadable from http://webinos.org

documentation as this contained implicit assumptions about prospective *webinos* users from the project team itself, making it a useful source of data from which the team could build personas.

In order to address the danger of taking repurposed data out of context, or introducing too much subjectivity into derived artifacts, design rationale techniques were used to justify explicitly any claims made. Because the amount of data in the documentation was substantial, the process of building personas from repurposed data was distributed among team members. The first step was to identify *factoids* from the design documentation, followed by affinity diagramming to cluster these according to potential persona behaviour. Each individual cluster formed the basis of a persona characteristic, and each group of clusters formed the basis of an individual persona. Each cluster was then structured systematically by aligning each characteristic with a model of argumentation. Each characteristic claim was built upon grounds, a warrant connecting the grounds to the claim, a qualifier stating the confidence in the claim being made, and any possible rebuttals to the claim; this model is described in more detail in [10], and proved extremely helpful in building useful design artifacts from repurposed data.

B. *Usability is not a priority*

Even if *webinos* was a Research & Development project, there was still underlying tension between the need for results and the need for research. It was, therefore, inevitable that usability design would clash with the design activities that they were intended to support. When this did happen, the contention led to some unexpected results, especially with respect to the time taken to build the context of use artifacts. For example, the team estimated that developing each persona would take half to a full working day; this was based on the number of data sources, and the time taken to both elicit factoids and carry out affinity diagramming. The team had assumed that partners would build each persona in a single sitting, because they knew affinity diagramming was best undertaken when people were most sensitised to the data, and building the argumentation model and associated narrative text was comparatively quick. What they found instead was that a number of personas were completed over the course of one to two *months*. This led to delays in specifying the activity scenarios and other dependent context of use artifacts, and, as a result, delays in identifying vulnerabilities that had not been found during the use case and requirement elicitation activities.

It was also found that team members failed to appreciate both the level of analysis necessary to cull factoids from the data sources themselves, and the time that this would take. Thus, as deadlines came closer, some team members identified factoids based only on a superficial analysis of data, rather than at the level of detail associated with a thorough qualitative data analysis. In hindsight, this was understandable: although the process of building personas was straightforward, the analysis and sense-making activities were difficult to explain using printed material and telephone conferences alone.

To give team members time to carry out the necessary data analysis activities, provide feedback on their work, and appreciate the contribution that the context of use description might play, the team held a two-day workshop one month before the context of use description delivery deadline. Preparing for the workshop led to the team members ensuring that their work was ready for validation. The event was also an opportunity to demonstrate how vulnerabilities could be derived from activity scenarios. After the experienced team members presented activity scenarios, and discussed possible vulnerabilities arising from the personas interaction with *webinos*, the less experienced team members became more confident and open to identifying possible threats and vulnerabilities evident in their own (and others') work.

C. *Technique misappropriation is easy*

Detailed guidelines, examples, and references were provided for team-members to help apply the described security and usability design techniques. Although these were written with the limited usability expertise of team members in mind, cases were still found where techniques were misappropriated unintentionally.

When writing the guidelines for building personas, the team explained that the purpose of the argumentation models was to help other team members better understand how persona characteristics were derived. Thus they were surprised that rather than using these characteristics to decrease stereotypical assumptions, some team members used them to achieve the opposite effect. In particular, they found that some personas were built by first writing the persona narrative and then using the argumentation models to justify assumptions that were made in the narrative; this effectively bypassed the data elicitation and affinity diagramming process altogether. These problematic personas were only identified during later reviews because the argumentation models used to underpin these personas were noticeably weaker than those that had been properly built from the data. In another case, a designer assumed that personas could be created by simply writing a narrative describing the expectations about a type of user he had in mind. When reviewing one of the personas created by this designer, the team drew attention to the fallacies underpinning several of the persona's characteristics, and walked through how the technique — and the argumentation models in particular — identified these fallacies, while also eliciting claims about the persona.

D. *Sustaining adoption through implementation requires creativity*

Looking back on the project, the results of the security and usability design activities showed some success. The misuse cases helped validate the *webinos* security architecture and, where these were not or only partially mitigated by the security architecture, recommendations had been made about how to address these. Almost all of these were only partially mitigated because multiple measures would be necessary, including the specification of more requirements before they could be implemented, as well as making APIs explicitly less permissive. For example, one automotive misuse case described how an attacker persona, fulfilling the role of a car-parking attendant, would be able to access music from another persona's devices by abusing the automatic login features and weak policy controls set by that persona. The proposed approach for dealing with this problem involved additional

Fig. 4. Consolidated concept map of requirements

authentication controls, and removing access to capabilities in automotive contexts where the team were confident that personas would not need them. It was, therefore, evident to those team members working on design activities that the dialectic between usability and security artifacts helped guide requirements elicitation and security design decisions.

While the team members continued to make use of this work when the implementation started, the security and usability design elements were noticeably less well used by other members of the project. Consequently, although general usability problems resulting from this analysis were addressed, such as documenting *webinos* concepts for use by application developers, usability problems with security mechanisms were largely forgotten in the rush to implement the core *webinos* architecture. This had two consequences.

First, many of the requirements associated with usable policy controls were made the responsibility of the development team building concept apps. This meant that, in addition to ongoing development work, they also needed to contend with the usability design of context-specific access control policies. For example, the *Kids in Focus* game was primarily created to demonstrate a cross-device web application between different collections of devices, because the young son played the game using *Helen*'s tablet device, while *Peter* played using the TV of his home-media system. In addition to demonstrating core *webinos* functionality, the game was also designed to demonstrate how conflicting privacy expectations might be managed. In particular, the application could access device geolocation data, to enable *Peter* to find out where *Helen*'s car was and how long before her car reached its destination. However, *Helen* was only prepared to disclose the time remaining and not her car's location. Unfortunately, because of time and resource constraints, the concept app team did not consider policy considerations a priority feature so, despite the importance of security and privacy considerations in the platform, these capabilities could not be demonstrated in the application's initial release.

To address this problem, team members provided a supporting role during concept app development. This involved providing advice on the security and privacy expectations of the game's users by describing how the personas might use the applications, and applying known security usability design guidelines [36] to indicate where authentication and authorisation decisions would and would not be necessary. Because there were continual discussions around the young child's interaction, the team decided to develop an additional persona (*Eric*) for *Helen*'s five-year old son. The data grounding this persona was derived from interviews with two *webinos* team members with young children. While less empirically grounded than the other personas, Eric proved invaluable during team discussions because almost none of the other team members were parents. The problem also subsequently led to design research into how context-sensitivity could be incorporated into *webinos*' access control framework. Arguably, the ability to have carried out this work would not have been possible had project team members not been engaged in the creation and use of personas. More information about this work can be found in [12].

Second, during the development of the core *webinos* architecture, vulnerabilities and threats that had been previously identified and described in detail were forgotten and 're-discovered'. Rather than review the work that had already been carried out, developers tried to solve these problems themselves, leading to sub-optimal implementation decisions. For example, creating a secure storage system — to keep personal data and credentials confidential — was consistently re-discovered by project members working on core functionality, context-awareness and the security architecture. Authentication requirements and solutions were also re-invented by concept app developers despite early specification work on the main system to solve the same problem.

The team dealt with this by identifying practical ways in which the usability design results could help the on-going development process without unduly impacting the implementation of core functionality. Time pressures meant that attempting to directly interface usability design activities with the Scrum practices would have been met with resistance. Therefore, the team supported on-going activities these practices would need to be aligned with; one of these involved synchronising the Scrum product backlog with the *webinos* requirements specification. This synchronisation was made possible by the release of the webinos design data described in Section III-E. Although there is little evidence to suggest that external users of *webinos* took advantage of the released data, the process of making the data more open increased its visibility throughout the project team.

To check whether important security and usability concerns had been properly addressed, the team helped the developers create concept maps [19] to make sense of the requirements — their relevance and priorities — and the connections between them and other design concepts. Each requirement was characterised as a single concept relating to its goal, and using pieces of paper for the requirement name and butcher paper as a surface, spatially arranged and associated with other concepts to infer the conceptual relationships between them. The development team members carried out this activity during project meetings when all team members were present in the same location. The team members found the exercise so useful that, despite other commitments, they decided to hold a subsequent workshop three weeks later to consolidate the requirements concept maps for each area; the consolidated concept map is shown in Figure 4, and the process itself is described in more detail in [13].

V. Conclusion and Lessons Learned

This paper described the difficulties faced designing security and usability into a non-trivial software system for a research and development project. In doing so, we believe we have presented the first study that shares insights, techniques, and outcomes of building a secure and usable system from the outset.

At first blush, many of the experiences described in Section IV seem unsurprising. Quality is often sacrificed in the rush to meet deadlines and, under such environments, latent conditions inevitably arise; these lead to errors and violations that cause the technique misappropriation problems described in Section IV-C [23]. However, labelling these experiences as corollaries of working on a risky project masks some potentially useful lessons. We summarise four lessons learned that practitioners can take away from our experiences designing and building *webinos*.

A. Learn to work with sub-optimal data and expertise

Even when security and usability are pushed to the front of a design process, it may be necessary to accept that high quality empirical data may either be unavailable, or not available in the quantity we would like. Moreover, the expertise required for collecting and analysing this data might not be readily available in climates where software is valued over models. Rather than spurning design artifacts grounded in reused, secondary, or assumption-based data, we need to acknowledge that imperfect data plays a valid role in security design. Moreover, as artifacts like personas become more visible in general security design texts, e.g. [28], we need to explore how design techniques can be better applied by non-experts, and how these techniques can improve the quality and presentation of the data we do have.

B. Security increases sensitivity to usability problems

Not only are security and usability both considered by R&D teams as secondary goals when developing systems, the qualities affecting security mechanisms are considered as tertiary. Fortunately, a by-product of applying security and usability design techniques together appears to be improved sensitivity to usability problems associated with security mechanisms. Therefore, we should ensure that these design activities are not carried out independently or are in tension with each other, and a supportive environment is available for reflecting on the interplay between security and usability values.

C. Designing for usability and security takes time

Personas, misuse cases, and attack trees may be the most visible output of *webinos*' usability and security design activities, but these cannot be created without intermediate design artifacts like affinity diagrams, and design rationale models. Not only are these intermediate artifacts necessary, designers also need time for self-reflection when building and applying them in security contexts. This is particularly important given the increasing use of security and usability research where agile and lightweight techniques are used, and where collaborating engineers are encouraged to *fake* process and focus on a final product [21]. While security and usability design techniques help to solve problems, designers also need to add their own expertise and understanding. As such, we would argue that we, as a community, need to extoll the virtues of critical thinking as a design tool, rather than relying exclusively on methods as a panacea.

D. Design research is a key element of designing of usability and security

As the work described in [12], [13] shows in more detail, security and usability design problems often require creative solutions. This study has shown that integrating usability design practices into a project at an early stage, and sensitising team members to some of the techniques necessary to carry out this research, lays the groundwork for carrying out the research necessary for identifying the causes, rather than the symptoms, of design problems as and when they arise. Moreover, as the results of the concept mapping exercise illustrated, such research also engages developers towards tackling their own design problems, while simultaneously sensitising them to the security and usability issues associated with them.

VI. Acknowledgments

The research described in this paper was funded by the EU FP7 *webinos* project (FP7-ICT-2009-05 Objective 1.2).

References

[1] A. Atzeni, C. Cameroni, S. Faily, J. Lyle, and I. Fléchais, "Here's Johnny: a Methodology for Developing Attacker Personas," in *Proceedings of the 6th International Conference on Availability, Reliability and Security*, 2011, pp. 722–727.

[2] C. Birge, "Enhancing research into usable privacy and security," in *Proceedings of the 27th ACM international conference on Design of communication*. ACM, 2009, pp. 221–226.

[3] J. Camenisch, S. Fischer-Hübner, and K. Rannenberg, *Privacy and identity management for life*. Springer, 2011.

[4] R. Chiesa, S. Ducci, and S. Ciappi, *Profiling hackers: the science of criminal profiling as applied to the world of hacking*. Auerbach Publications, 2009.

[5] A. Cockburn, *Writing Effective Use Cases*. Addison-Wesley, 2001.

[6] J. Dowell and J. Long, "Towards a conception for an engineering discipline of human factors," *Ergonomics*, vol. 32, no. 11, pp. 1513–1535, 1989.

[7] S. Faily and I. Fléchais, "Barry is not the weakest link: eliciting secure system requirements with personas," in *Proceedings of the 24th BCS Interaction Specialist Group Conference*, ser. BCS '10. British Computer Society, 2010, pp. 124–132.

[8] S. Faily and I. Fléchais, "Designing and aligning e-science security culture with design," *Information Management and Computer Security*, vol. 18, no. 5, pp. 339–349, 2010.

[9] ——, "To boldly go where invention isn't secure: applying Security Entrepreneurship to secure systems design," in *Proceedings of the 2010 New Security Paradigms Workshop*. ACM, 2010, pp. 73–84.

[10] ——, "Persona cases: a technique for grounding personas," in *Proceedings of the 29th international conference on Human factors in computing systems*. ACM, 2011, pp. 2267–2270.

[11] ——, "User-centered information security policy development in a post-stuxnet world," in *Proceedings of the 6th International Conference on Availability, Reliability and Security*, 2011, pp. 716–721.

[12] S. Faily, J. Lyle, I. Fléchais, A. Atzeni, C. Cameroni, H. Myrhaug, A. Göker, and R. Kleinfeld, "Authorisation in Context: Incorporating Context-Sensitivity into an Access Control Framework," in *Proceedings of the 28th British HCI Group Annual Conference on People and Computers*. British Computer Society, 2014, to Appear.

[13] S. Faily, J. Lyle, A. Paul, A. Atzeni, D. Blomme, H. Desruelle, and K. Bangalore, "Requirements sensemaking using concept maps," in *Proceedings of the 4th International Conference on Human-Centered Software Engineering*, ser. HCSE'12. Springer-Verlag, 2012, pp. 217–232.

[14] J. Hobek, "The innovation design dilemma: some notes on its relevance and solution," in *Innovation: a cross-disciplinary perspective*, K. Grønhaug and G. Kaufmann, Eds. Norwegian University Press, 1988.

[15] Institute for Information Infrastructure Protection, "Report: 1st Software and Usable Security Aligned for Good Engineering (SAUSAGE) Workshop," http://www.thei3p.org/events/sausage2011.html, April 2011.

[16] ISO, "ISO 9241-11. Ergonomic requirements for office work with visual display terminals (VDT)s - Part 11 Guidance on usability," Tech. Rep., 1998.

[17] A. Martin, J. Davies, and S. Harris, "Towards a framework for security in escience," in *Proceedings of the IEEE e-Science 2010 Conference*, 2010.

[18] NeuroGrid Consortium, "NeuroGrid website," *http://www.neurogrid.ac.uk/consortium.htm*, 2005.

[19] J. D. Novak and D. B. Gowin, *Learning How To Learn*. Cambridge University Press, 1984.

[20] OWASP Foundation, "Open Web Application Project (OWASP) web site," http://www.owasp.org, July 2014.

[21] D. L. Parnas and P. C. Clements, "A rational design process: How and why to fake it," *IEEE Transactions on Software Engineering*, vol. 12, no. 2, pp. 251–257, 1986.

[22] PrimeLife Consortium, "PrimeLife web site," http://www.primelife.eu, November 2011.

[23] J. Reason, *The Human Contribution: Unsafe Acts, Accidents and Heroic Recoveries*. Ashgate, 2011.

[24] M. B. Rosson and J. M. Carroll, *Usability engineering: scenario-based development of human-computer interaction*. Academic Press, 2002.

[25] M. A. Sasse, S. Brostoff, and D. Weirich, "Transforming the 'weakest link' – a human/computer interaction approach to usable and effective security," *BT Technology Journal*, vol. 19, no. 3, pp. 122–131, July 2001.

[26] B. Schneier, "Attack Trees," *Dr. Dobb's Journal*, vol. 24, no. 12, pp. 21–29, 1999.

[27] K. Schwaber, *Agile Project Management with Scrum*. Redmond, WA, USA: Microsoft Press, 2004.

[28] A. Shostack, *Threat Modeling: Designing for Security*. John Wiley & Sons, 2014.

[29] G. Sindre and A. L. Opdahl, "Eliciting security requirements with misuse cases," *Requirements Engineering*, vol. 10, no. 1, pp. 34–44, 2005.

[30] J. Ure, F. Rakebrandt, S. Lloyd, and A. A. Khanban, "Usability, the tri-wizard challenge: Recurring scenarios in the design of a healthgrid portal," in *Proceedings of the 2008 Conference on Human System Interactions*. IEEE Computer Society, 2008, pp. 298–305.

[31] webinos Consortium, "User Expectations for Security and Privacy Phase 2," http://webinos.org/2011/11/01/webinos-repot-user-expectations-of-security-and-privacy-phase-2/, November 2011.

[32] ——, "User expectations on privacy and security," http://webinos.org/content/webinos-User_Expectations_on_Security_and_Privacy_v1.pdf, February 2011.

[33] ——, "Updates on Scenarios and Use Cases," http://www.webinos.org/wp-content/uploads/2012/06/D2.4-Updates-on-Scenarios-and-Use-Cases.public.pdf, May 2012.

[34] ——, "webinos source code," https://github.com/webinos, January 2012.

[35] webinos Consortium, "webinos web site," http://webinos.org, March 2012.

[36] K.-P. Yee, "Guidelines and strategies for secure interaction design," in *Security and Usability: Designing Secure Systems that People Can Use*, L. F. Cranor and S. Garfinkel, Eds. O'Reilly Media, 2005, pp. 247–273.

[37] M. E. Zurko and R. T. Simon, "User-centered security," in *Proceedings of the 1996 New Security Paradigms Workshop*. ACM, 1996, pp. 27–33.