systems

Article

RAD-XP: Tabletop Exercises for Eliciting Resilience Requirements for Sociotechnical Systems

Stephen L. Dorton *, Emily Barrett, Theresa Fersch, Andrew Langone and Kelly J. Neville

The MITRE Corporation, McLean, VA 22102, USA; kneville@mitre.org (K.J.N.)
* Correspondence: sdorton@mitre.org

Abstract: Despite noble intentions, new technologies may have adverse effects on the resilience of the sociotechnical systems into which they are integrated. Our objective was to develop a lightweight method to elicit requirements that, if implemented, would support sociotechnical system resilience. We developed and piloted the Resilience-Aware Development Exercise Protocol (RAD-XP), a method to generate tabletop exercises (TTXs) to elicit resilience requirements. In the pilot study, this approach generated 15 requirements from a one-hour TTX, where the majority of requirements were found to support resilience. Participants indicated via survey that RAD-XP was effective and efficient, and that they would want to use RAD-XP regularly throughout the agile development process. We discuss future research and development to refine this approach to eliciting resilience requirements.

Keywords: resilience; sociotechnical systems; tabletop exercise; requirements elicitation

Citation: Dorton, S.L.; Barrett, E.; Fersch, T.; Langone, A.; Neville, K.J. RAD-XP: Tabletop Exercises for Eliciting Resilience Requirements for Sociotechnical Systems. *Systems* **2023**, *11*, 487. https://doi.org/10.3390/systems11100487

Academic Editors: Christina Rusnock and Michael E. Miller

Received: 24 July 2023
Revised: 15 September 2023
Accepted: 20 September 2023
Published: 23 September 2023

1. Introduction

Advanced technologies are increasingly critical to successful operations in high-consequence sociotechnical systems, which can be problematic when despite gains in efficiency, new technologies adversely affect the resilience of that system. While multiple definitions exist for sociotechnical systems (also referred to as "work systems"), they typically refer to some collection of humans, organizations, agents, technology, and/or information operating together toward a common goal [1–3]. We align these common descriptions with the cognitive systems triad [4] and define a sociotechnical system as a collection of humans (individuals and/or teams), technology, and work (procedures, policies, etc.) that work towards a common goal. Modern high-consequence sociotechnical systems evolve over time and in response to challenges, allowing them to achieve a certain level of resilience, or ability to adapt to surprises or challenging conditions with adaptive capacity [5].

When we modernize work organizations or introduce new technologies, our goal is often to improve efficiency or performance. We often focus on those goals and assume the technology will be adopted into use; however, any technology injected into the complex, time-honed dynamics of an established sociotechnical system is unlikely to be prepared for participating effectively in those dynamics. As such, it is unfortunately common that the introduction of new technologies to an existing sociotechnical system can have adverse unintended effects [6–8].

An underlying theme in many of these instances of unintended outcomes is that the new technology degrades the resilience of the sociotechnical system upon its introduction [9]. Although there are various definitions, a system's resilience can be defined as its ability to adjust functioning prior, during, or after various events to sustain operations and achieve goals [10]. An ill-prepared technology may interfere directly with evolved mechanisms for resilience, making the work system vulnerable. Such a technology is even more likely to have disruptive effects when developed to improve efficiency, security, control, and predictability, or a specialized capability, as these goals can trade off with the goal of resilience. For example, an automation designed to prevent medical professionals from ordering patients two or more medicines with hazardous interactions was discontinued after evaluations showed multiple cases where the technology was brittle to patients' circumstances, and caused an unacceptable delay in producing medications for patients. In at least one case, this system led to a break in a life-saving medication regimen for a patient. While designed to increase safety by preventing fatal medical errors, a lack of resiliency in dealing with patients' contextual factors led to the technology having an adverse effect [11].

Neville et al. [9] introduced the concept of Resilience-Aware Development (RAD) as a mechanism to reduce the likelihood of a new technology degrading the resilience of the sociotechnical system by interfering with responsiveness and adaptivity. That is, RAD focuses on considering resilience throughout the system development lifecycle, from conceptualization through system deployment and sustainment, and every step in between. Neville et al. [9] built the RAD paradigm around the Transform with Resilience during Upgrades to SocioTechnical Systems (TRUSTS) Framework of Work System Resilience, which identified five key factors of resilient sociotechnical systems, paraphrased as:

- Shared Demand and Deviation Awareness: Each unit in the system is aware of changes from the norm and/or demands that require responses;
- Progressive Responding: The system engages in continuous sensemaking and decision making to align responses to demands;
- Response Coordination: The system has tacit and explicit mechanisms to support omnidirectional and adaptive coordination of resources;
- Maneuver Capacity: The system has a diversity of means, and can act in novel ways, to achieve goals;
- Guided Local Control: Higher work units provide flexible guidance to front-line units, who have the flexibility and authority to respond to incidents.

Effective system development relies upon high-quality requirements [12], which are supported by sound Requirements Engineering (RE) practices and tools. The remainder of this manuscript focuses on an RE tool to iteratively elicit requirements that will prepare a novel technology to participate in the resilient work operations of high-consequence work systems, (i.e., "resilience requirements"). The RAD Exercise Protocol (RAD-XP) builds upon the TRUSTS framework and is used to design and conduct lightweight Tabletop Exercises (TTXs) throughout a technology's development lifecycle to proactively stress the technology and elicit resilience requirements in the form of user stories.

This paper is structured as follows. For the remainder of the Introduction, we discuss challenges to RE in the context of agile development, present arguments for how TTXs can overcome many of these challenges, and compare and contrast RAD-XP with related methods. In Section 2, we provide an overview of RAD-XP, illustrated with examples and lessons learned from a pilot study, where appropriate. In Section 3, we describe a pilot study conducted with a novel healthcare technology and provide results from mixed-methods research on the outcomes of the pilot TTX. In Section 4, we highlight key findings, discuss limitations of the pilot, and cast a vision for future research and development of RAD-XP.

1.1. Agile Development

A majority of software development projects employ agile development, or at least some form of hybrid-agile method [13]. Agile development, as expressed in the Agile Manifesto [14], aims to improve software development by shifting emphasis from generation and maintenance of documentation, to collaboration with customers and delivering software early and often. Arguably the most substantial paradigm shift, however, is the idea that developers should embrace and respond to change, rather than seeking to follow an initial plan. Scrum is the most popular agile development method [15], where development is generally accomplished in short "sprints" of effort, although the exact process varies from team to team [16]. Agile development, along with other approaches such as User Experience (UX) design, has become the standard practice in the last two decades [17].

The adoption of agile development methods provides an opportunity for RAD tools such as RAD-XP that progressively reveal emergent requirements over the course of a technology's development. The ways a technology design can interact with work system operations are difficult to predict in advance [8]. As the prototype matures, the possible interactions become increasingly apparent and predictable. RAD-XP facilitates the evaluation of these possible interactions, identifies resilience-related opportunities and risks, and generates resilience requirements for incorporation into future agile iterations.

Despite its popularity, there are several challenges to conducting RE in agile development processes. Kelly [18] noted that stakeholders often have trouble seeing beyond the current situation and may be unwilling to be involved in the process. This lack of stakeholder involvement can exacerbate issues with developers who do not sufficiently understand the problem domain, and who may not even know what questions to ask stakeholders if they were to be involved [19]. Kelly [18] also noted that it can be difficult to iteratively develop requirements as the technology matures over each sprint. Previous studies have noted a lack of requirement-elicitation guidelines for agile development, where 90% of surveyed participants said they would like a guide or process for eliciting (non-functional) requirements [20].

1.2. Tabletop Exercises & Wargames

Tabletop Exercises (TTXs), wargames, and gamification more broadly, provide various benefits toward effective RE and resilience engineering efforts [21]. TTXs are a type of relatively low-fidelity simulation involving human participants, where participants are presented with a situation or world state in each turn of play, whereby they can deliberate and test various ideas of how to best deal with the situation with minimal effort or consequence (e.g., a TTX of how to respond to a wildfire is cheaper and safer than actually starting and putting out a wildfire for training purposes). Participants usually submit a response or "move" they would make in each turn of play to a facilitator, whereby the facilitator presents participants with the next situation to respond to. TTXs are usually generated to explore or test how to respond to an adverse situation (e.g., warfare, natural disaster, or other emergency). That is, the primary inputs of a TTX are some scenario or situation of interest, and research questions to be answered. Participants in TTXs are typically the domain experts, stakeholders, or end users of a technology that the TTX aims to explore (e.g., firefighters and forestry experts would be likely participants in a wildfire TTX). The outputs of TTXs vary considerably, but are usually insights and answers to research questions.

Caffrey [22] notes that the primary difference between a TTX and a wargame is that a wargame involves a thinking adversary, while a TTX can be played without one (the adversarial elements are typically developed and scripted prior to the TTX). For the sake of simplicity, we acknowledge this distinction, but use the term TTX throughout the remainder of this manuscript to be inclusive of both proper wargames and TTXs (a subset of wargames). TTXs have been historically used to develop and test military tactics [22] but have more recently been adapted for use in systems engineering contexts. For example, TTXs have been used to develop novel concepts for system employment [23], assess employment and adoption of novel technologies like unmanned systems [24,25], elicit requirements for user interfaces [26], and develop and refine measures of cyber resiliency to support test and evaluation activities [27]. TTXs are a mechanism to proactively expose a novel technology to an envisioned version of a future sociotechnical system, i.e., something advocated as being critical to better development and acquisition of intelligent systems [28,29].

The use of TTXs in the systems engineering process is a high-impact, low-cost approach to developing contextually rich insights for the requirements, designs, and testing of systems, at even early phases of the development lifecycle where there is no working code [26]. Different fidelity TTXs (e.g., paper-based tabletop vs. digital simulations) have been shown to be more effective at achieving exploratory or confirmatory objectives, respectively [23]. Further, de Rosa and Strode [24] have shown that knowledge-acquisition analytical games (a type of game focused on knowledge elicitation) are both effective and efficient: they allow for elicitation of qualitative and quantitative knowledge in a simple and rapid manner.

TTXs also provide "soft benefits" to participants, primarily through collaborative discussions that provide opportunities to interact and share perspectives with people from different backgrounds [25], and contribute to a shared mental model among the participants who are often colleagues [30]. Similarly, Lombriser et al. [31] noted that TTXs implicitly helped coordinate with stakeholders. Finally, Hahn [32] discussed how a TTX not only generated better, contextually-driven requirements, but also made participants aware that their measures of performance needed adjusting.

More broadly speaking, gamification, or the use of game design elements in non-game contexts (e.g., training applications or RE), has been demonstrated to increase motivation and engagement through a variety of underlying mechanisms [33]. More specifically, TTXs exhibit several gamification mechanisms such as goals, narratives, and time limits, to name a few. As such, increased motivation and engagement have been reported as an outcome of using gamified methods such as TTXs. References [23,26] showed that participants in analytic games felt involved in gameplay, had fun participating, and found the games useful. Similarly, others have reported a positive social aspect of participating in analytic games with colleagues, and preferred gaming over other means for knowledge elicitation such as surveys [25]. Lombriser et al. [31] proposed a gamified framework for RE, and noted that participants reported subjectively feeling more motivated, and objectively exhibited increased participation rates. Further, the gamified RE methods generated significantly more user stories, which had significantly higher quality, than user stories generated using non-gamified methods.

1.3. Related Methods

While RAD-XP was developed predominantly through the lenses of resilience engineering and wargaming, it is worth acknowledging that it bears similarities to other RE methods. It is important to acknowledge these similarities, as well as key differences between RAD-XP and other conceptually similar RE methods.

RAD-XP bears similarities to the Scenario Requirements Analysis Method (SCRAM) [34,35] in several ways. SCRAM is a four-step process that generates requirements for an early prototype and employs a participatory process to prioritize requirements where tradeoffs exist. Like RAD-XP, SCRAM evaluates the developed technology in the context of use, and also relies on a mixture of end users and developers as participants. A key difference, however, is that SCRAM relies on an existing prototype or artifact so that participants can react to a preliminary design [34]. RAD-XP is designed to be flexible and can be conducted regardless of whether any artifact or prototype exists (as was the case in the pilot). This allows RAD-XP to be conducted earlier in the development lifecycle, to proactively explore resilience requirements before any implementation takes place.

RAD-XP also shares common principles with obstacle- and fault-based RE methods. Such approaches generally work by refining high-level system goals into subgoals, and then identifying obstacles to achieving those goals [36] through a formalized model (e.g., fault tree) with semantics and logic. Requirements are elaborated as a product of top-down and bottom-up processes to overcome identified obstacles (e.g., by goal or agent substitution). Both RAD-XP and obstacle-based approaches focus on the technology as part of a sociotechnical system (i.e., technology, people, and the environment/work), and both assume the technology must operate under non-ideal conditions [36]. Although they are aligned in principle, RAD-XP and obstacle-based methods are divergent in practice. While RAD-XP is intended to be a lightweight method that does not require specialized expertise, even a simplified implementation of obstacle analysis [37] involves formalized logic and modeling that is likely to elude many practitioners. Further, RAD-XP only aims to elicit requirements, while obstacle-based methods tend to lend themselves to quantitative analyses for prioritizing requirements under constraints (e.g., budget) [38]. Finally, RAD-XP requires end users to participate, while such participation does not appear to be a fundamental part of obstacle-based approaches.

Various other RE methods bear some similarity to RAD-XP. Karolita et al. [39] reviewed RE methods that relied on user personas (and their associated use cases), but found it time consuming to collect data from stakeholders with low motivation to participate. Other methods have looked at templates to help developers elicit requirements [40] to support trust, usability, etc., but not resilience. Others have looked at data-driven approaches to elicit user requirements from online sources such as blogs and reviews [41]; however, these methods are purely reactive, and rely on the system to have already been developed and deployed. While RAD-XP is not entirely novel, as it shares various principles and practical facets with existing RE methods, there are noted differences with each of these related methods, and a gap exists for participatory RE methods focused on sociotechnical resilience.

1.4. Challenges & Potential Mitigations

In summary, we face several challenges in eliciting requirements to support work system resilience. These challenges arise from different sources, where some challenges are due to the complex nature of resilience engineering, and some are associated with RE more generally, while others are borne specifically from agile development methods. We also discussed the numerous advantages to using wargames in the context of RE (or systems engineering more broadly). We present a non-exhaustive summary of these challenges, along with some arguments for how wargames are suited to mitigate or overcome said challenges, in Table 1.

Table 1. Overview of agile requirements engineering challenges and TTX mitigations.

Challenge	TTX Mitigation
Software development is increasingly a distributed activity [42], and RE can be difficult when teams are distributed [15,20].	TTXs are flexible and can be readily executed in geographically- [43] and temporally-distributed settings [44].
RE in Agile relies heavily on direct engagement with end users and stakeholders; however, it can be difficult to get stakeholders involved in RE [18,39,45].	Research has shown that busy stakeholders were highly motivated to participate in TTXs [26], and that such gamified methods for RE result in increased participation and better user stories [31,32].
There is a lack of a common language to support the different mental models of users, stakeholders, and developers [46]. Users often struggle to articulate requirements [47,48], while developers and requirements engineers often do not know the right questions to ask users [19], despite needing domain knowledge to be successful [45,49].	Hosseini et al. [50] found that involving large and diverse crowds in RE increased the accuracy, relevance, and creativity of resultant requirements. Military participants in a TTX reported that participation allowed them to meet and collaborate with their foreign counterparts that they had not met before, illustrating how TTXs foster collaboration across diverse groups of participants [25].
The rapid and iterative nature of Agile creates challenges for RE, where traditional RE methods may be too slow to be impactful [26], and requirements must be iteratively developed as the technology matures with each sprint [18].	Dorton et al. [26] demonstrated the ability to design and execute TTXs rapidly enough to have early impact in an Agile lifecycle. TTXs can be performed iteratively over periods of time [44], such as during sprints.
Stakeholders and end users can have difficulty seeing beyond the current situation into the future, and in thinking concretely about envisioned future states [18].	Scenario-based requirements engineering approaches (including TTXs) have been shown to generate detailed discussion by grounding argument and reasoning in a specific scenario [12,34]. Depending on the scenario fidelity, TTXs may enable observational research, rather than discussion-based research that is more speculative [26].

2. RAD-XP

Based on the benefits noted in Table 1, we developed and pilot-tested the Resilience Aware Development Exercise Protocol (RAD-XP), a structured and adaptable method to design, execute, and analyze lightweight TTXs to elicit resilience requirements. The standard RAD-XP TTX is completed in less than 90 min (including introductions and a hot wash) and plays through a scenario with multiple injects. The scenario is the overarching "scene" in which the exercise is played and provides high level context to the use of the technology. The injects are simulated events that are designed to stress specific resilience factors of the technology and the overall work system, and require players to respond to via a set of prompts (i.e., questions).

Figure 1 provides an overview of the RAD-XP pilot that was conducted in three phases, discussed in the following sections. Although we have only conducted a single pilot exercise, we have designed RAD-XP to be employed regularly throughout the agile development process (e.g., with each sprint or each delivery, depending on needs), where TTXs generate resilience-focused user stories to populate the backlog. The RAD-XP is designed to be planned and executed by software development teams, although we have facilitated this initial pilot for the participating team.

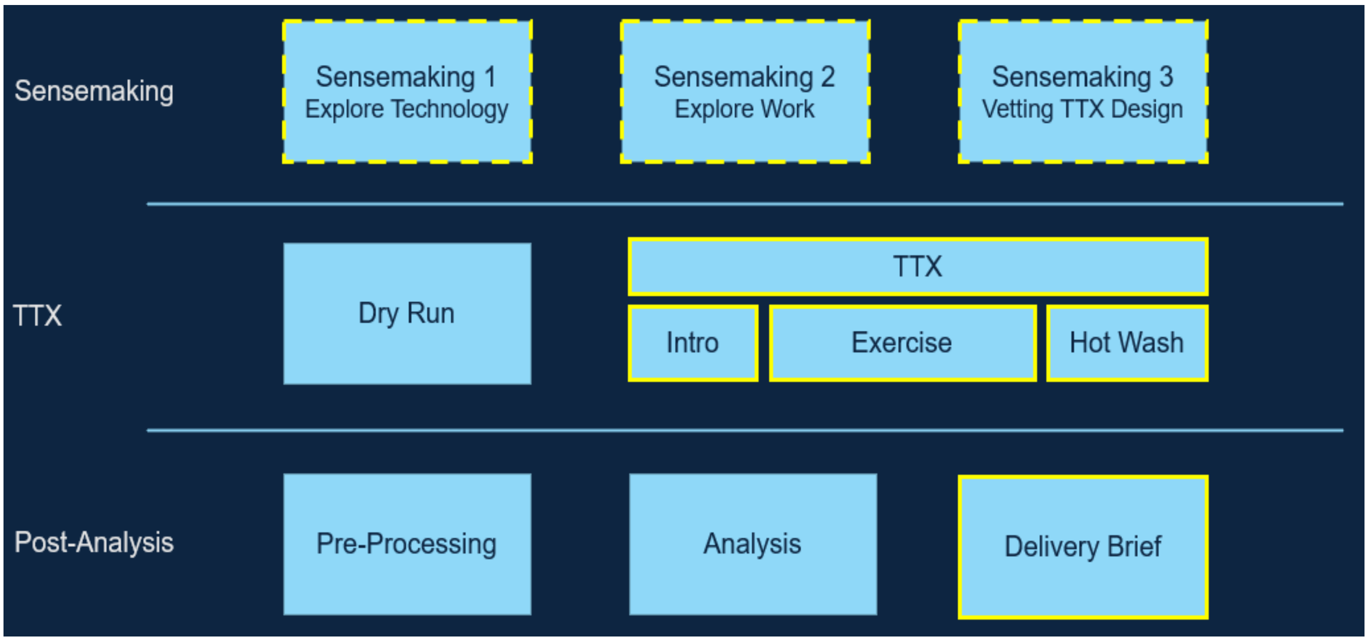

Figure 1. RAD-XP Workflow. Activities with yellow borders require involvement of the entire development team, while activities with dashed yellow borders require only 1–2 key personnel from the development team.

2.1. Sensemaking Phase

The longest and most labor-intensive phase of the RAD-XP is a series of meetings and analysis to make sense of the technology being developed (the 'what'), and the contexts in which it will be used (the 'who,' 'how,' 'when,' 'where,' and 'why'). The first two meetings (Sensemaking 1 & 2) focus on eliciting knowledge from the development team to enable design of the TTX. While RAD-XP is focused on requirements to support the resilience of the larger sociotechnical system (not to support the individual user), the sensemaking phase closely resembles activities common in UX and other user-centered design practices [51,52]. Only the Project Lead(s) and Technical Lead(s) of the development team are involved in this phase, although more are welcome, so long as those involved can answer specific sensemaking questions:

- Purpose: Why is the technology being funded/built? What problem does it solve?
- People: Who uses the technology? Who else will interact with it (maintainers, administrators, etc.)? What are their goals?
- Form and Function: How does the technology work? What are the major functional components? How do users interact with it?
- Work: What missions or task threads will the technology be used in? What policies or procedures might govern technology use in work?

- Environment: Where is the technology used? When is the technology used? Will it be used under time pressure?
- Other Technology: What other technologies will this technology interact with (upstream or downstream)?

The TTX is designed based on the responses to these sensemaking questions. This involves designing the scenario (the overarching scene in which the TTX takes place) and the injects (events that stress the resiliency of the system and drive discussion during the TTX). We have designed nearly a dozen different archetypical (i.e., standardized, but technology-agnostic) scenarios that add global constraints (i.e., constraints or conditions such as resource shortages or extreme operational tempos, that last for the entirety of the TTX), although we recommend that the initial TTX for a given technology be a "sunny day" scenario without any such global constraints (i.e., where the injects are the only issues to respond to).

While scenarios set the scene and provide global constraints, injects are individual events within a given scenario. Injects can, for all practical purposes, be considered the "turns" in the TTX, where each inject is a new challenge to overcome. Designing injects is arguably the most critical step because it is the point at which the TTX becomes focused explicitly on resilience. At a high level, injects generally align with Klein's recommendations for making training scenarios cognitively difficult [53]: Adding ambiguity, misleading, or missing information; violating expectancies; time pressure; or giving goals or priorities that are conflicting or readily made obsolete by events. That is, they stress the resilience of the sociotechnical system in which the technology resides. At a more granular level, injects are developed using a matrix with a vignette as the column (elicited from sensemaking questions), and the factors and subfactors from the TRUSTS framework as rows [9]. The team works though the vignette, identifying what types of injects might test different resilience factors or subfactors (e.g., an inject where an upstream data feed is offline would stress the shared demand and deviation awareness of the work system).

The final step (Sensemaking #3) is to meet with leaders of the development team to gain concurrence on the utility and plausibility of the scenario. Proposed injects should not be discussed during Sensemaking #3 because the project leaders participate in the TTX and should not have knowledge of specific injects prior to the TTX itself.

2.2. Exercise Phase

The exercise phase is a short phase that requires a relatively low level of effort (completed within a single week). The first step is to translate the designed scenario and injects into a Master Scenario Event List (MSEL) to prepare for a dry run of the TTX. The MSEL is an administrative artifact, typically in the form of a spreadsheet, containing the step-by-step plan for the exercise, including start and stop times for each step, assignments of who would lead the step, and any other clarifying notes. A slide deck for facilitating the TTX also must be prepared, which includes front matter (e.g., overviews of both resilience engineering and the technology being explored), as well as the scenario overview and injects.

Teams then conduct a dry run exercise without participants. During the dry run, a team member should role play as an obstinate participant (often referred to as "red teaming") to force the team to plan for various responses and adaptations. In the pilot TTX, the dry run allowed us to proactively identify substantial issues (e.g., poorly written questions that were confusing or would limit responses) as well as simpler issues such as typos and grammatical errors in slides. These issues were adjudicated before the actual exercise.

The technology development team, users, and stakeholders then participate in the actual exercise. A facilitator leads the event by progressing though the MSEL, encouraging discussion on responses to injects. Another facilitator take notes and contributes to discussion with refining questions when appropriate. Facilitators execute the TTX by presenting the slide deck, starting with a slide for the scenario, and then a slide for each of the injects. Inject slides contain the description of the inject, and 3–5 questions prompting participants to describe how they would respond to the inject. In the pilot, we did not have participants answer each question specifically, but instead highlighted questions that were most important from a resilience standpoint and let discussion flow as naturally as possible. At the conclusion of the exercise, the facilitator runs a hot wash discussion where participants enumerate key takeaways from the TTX.

2.3. Analysis Phase

The analysis phase is the shortest phase (completed in a single day) and requires arguably the lowest level of effort. The overarching effort of the analysis phase is to translate participant discussion from the TTX into user stories, or more specifically, to map relevant utterances from the transcript of the exercise to slots in a user story template. RAD-XP uses the Connextra template because most development projects are agile [13], and most agile projects use the Connextra template [54]. Although specific verbiage varies across sources, this popular template generally takes the form of a simple statement with three slots to be filled in: As a [user], I want to [goal], so I can [justification] [55,56]. The 'user' slot identifies who the story pertains to, which could be a user, a maintainer, an administrator, or other party. The 'goal' slot describes the action the user wants to take, which implies a functionality in the technology. Finally, the 'justification' slot provides a rationale of why the user wants to take the action, usually framed as a benefit.

Elicitation of user stories is a simple process of visually scanning a transcript for certain flags (i.e., words likely associated with one of the slots in a user story). First, the transcript is scanned for any time an entity is mentioned, which is a flag for a user. This may require some contextual understanding, as shown in the example in Table 2, where knowing the participant was a Chief Medical Information Officer (CMIO) was critical to identifying the proper user in the resultant user story. Once a flag for a user was found, adjacent content in the transcript is searched to find flags for the other two slots. Users often appear as proper nouns, although "I" is commonly used because the participants place themselves in the scenario. Function flags are often verbs, describing actions to be taken. Finally, rationales can be more difficult to find, although certain phrases such as "because", "so" and "in order to" are common.

Table 2. User story elicitation example.

Annotated Example Transcript	User Story Slots
"If that happened, I would want to know if anything changed, in terms of outcomes, you know, between now and the last time, to make sure we should still be using it."	User: As a CMIO [1]...
	Function: I want to know if outcomes have changed since the last periodic review...
	Rationale: So I can make an informed decision about whether we should be using the tool.

[1] The participant had identified themselves as a retired Chief Medical Information Officer (CMIO), and was playing the exercise primarily from that viewpoint.

As shown in Table 2, the flag "I" was found on the first pass, but within the same sentence were the flags for the associated function and rationale. It should be noted that sometimes only two of the three slots are explicitly stated in the transcript (e.g., somebody might state which user needs to do what specific function, but not explicitly state the rationale or benefit). In these cases, the user story is still recorded, and future work with end users can help to fill in any missing slots.

3. Pilot Study

3.1. Methods

We conducted a pilot study of the RAD-XP using the methods described in Section 2. For this pilot we worked with a team developing a medical decision aid that determines the fitness of both populations and individual patients for obstetric (OB) telehealth service. The technology includes a user dashboard, an underlying synthetic population model, and different models to predict telehealth effectiveness based on clinical factors (e.g., blood pressure) and Social Determinants of Health (SDOH; e.g., access to broadband internet). The goal of the technology was to make healthcare more equitable by increasing telehealth options where appropriate.

The participants (*N* = 7) in the pilot event consisted of four members of the project team (two healthcare researchers and two software developers), the sponsor of the research (who was also a physician), and two domain subject matter experts. Both domain experts were physicians with experience in the relevant field of medicine, one of which was also a CMIO who oversaw the adoption of decision support technologies at their hospital. In summary, the participants consisted of two healthcare researchers, two software engineers, and three physicians (i.e., representative end users of the technology).

A simple "sunny day" scenario was used for this initial pilot, which assumed normal, intended use of the technology, with no significant issues predicted or reported. Approximately a dozen injects were developed using the inject matrix and the process described in Section 2: the RAD-XP team considered the general scenario of a physician using the technology to determine patient fitness for OB telehealth, and went row-by-row through the inject matrix (where each row represents a resilience enabler) to brainstorm an inject that would undermine that resilience enabler. Therefore, each inject was mapped to one or more resilience factors, and was plausible in the context of the overarching scenario. From this larger set, the following six injects were selected for use in the pilot TTX (the associated resilience factors from the TRUSTS framework are shown in parentheses):

- Inject 1: You are tasked with assessing the outcomes of a pilot study to measure the impact of OB telemedicine on a population (Response Coordination);
- Inject 2: The underlying criteria SDOH change, and you need to determine how this affects the technology's models, and your patients' qualifications for OB telehealth (Shared Demand and Deviation Awareness);
- Inject 3: Due to the changes in SDOH criteria, you need to override the systems' outdated models to manually change which patients qualify for OB telehealth (Guided Local Control);
- Inject 4: You would like to re-examine a current OB telehealth patient's fitness for OB telehealth after their living situation (and therefore SDOH) changed (Response Coordination);
- Inject 5: A patient's data at a face-to-face appointment is inconsistent with what was logged during telehealth appointments (Progressive Responding);
- Inject 6: Medical students want to access and share deidentified data with another research hospital to collaborate on a paper about equity in OB telehealth (Guided Local Control).

Each of these injects was followed up with 3–5 prompting questions to give the participants the chance to explain how they would respond to the given inject. The process to develop these prompting questions was not entirely standardized, but generally focused on how the participant would interact with the technology. Questions typically focused on how they would respond to the inject, why they would respond that way, what they would need to be successful, who they would coordinate with, and other contextual cues or factors that may drive adaptations in response to the inject. Although our intent is for development teams to conduct this procedure themselves in the future, it was advantageous to have an independent party (the RAD-XP team) without a vested interest in the performance of the technology write the injects and prompts. As such, we recommend that users of RAD-XP invite independent third parties to participate in inject development wherever possible, as development teams may be blind to potential resilience failures of their own technologies.

After the TTX was complete and user stories had been generated, a team of three experts in resilience engineering rated the resultant user stories to determine whether they supported work system resilience or not (i.e., they were functional or non-functional requirements with no foreseeable impact to resilience). Each rater reviewed the set of user stories and provided a binary rating (1 = supports, 0 = does not support) for each of the resilience factors enumerated in the TRUSTS framework [8]. These ratings were not mutually exclusive, meaning that a single user story could be mapped to multiple resilience factors. A simple majority consensus method was used to adjudicate ratings, where each rating was declared based on whether at least two of three raters agreed.

Finally, we administered two surveys: one immediately following the TTX (before the results had been generated), and the other immediately following the results delivery brief where the participating team was presented with the results of the TTX. These surveys were predominantly focused on assessing participant perceptions on the relative costs and benefits of conducting the RAD-XP-generated TTX as a tool for RE, and how to improve the RAD-XP going forward. Although we can objectively measure the outputs of the TTX in the number of user stories generated, we have no metric against which to determine effectiveness or efficiency (e.g., is 10 user stories good or bad?). Therefore, subjective feedback was critical to gleaning meaningful insights from the measured outcomes of the pilot.

The surveys included a set of Likert items on a seven-point scale. Participants could register their level of agreement as one of the following: strongly disagree, disagree, slightly disagree, neutral, slightly agree, agree, or strongly agree. Although surveys used a seven-point scale, the surveys were exploratory and qualitative in nature. That is, we did not seek to conduct any quantitative analysis (descriptive or inferential)—we only sought to glean qualitative insights on participant perceptions. The items in each of the two surveys were largely unique, although two items were present in both surveys to assess whether perceptions had changed after seeing the outcomes of the TTX. These surveys also included free-text responses to elicit contextual feedback that could not adequately be captured by Likert items or other response formats. These questions were aligned with the objectives of the surveys, and generally aimed to assess the perceived effectiveness of the TTX and mechanisms for improving future iterations of RAD-XP.

3.2. Results

3.2.1. User Stories

The stated objective of the TTX was to generate user stories that, if implemented, would support work system resilience. Toward this end, the pilot TTX generated 15 user stories in total. For 10 of the 15 user stories (67%), raters came to consensus on the specific mechanism(s) by which resilience was supported, or that it supported resilience in a general sense, not captured by the TRUSTS framework (i.e., raters could map the user story to one or more specific factors in the TRUSTS framework or declared it as generally supportive of resilience). There were three user stories where raters agreed that they supported resilience in some way but could not come to consensus on a specific factor. When accounting for these additional user stories without an identified mechanism, 13 of 15 (87%) user stories supported work system resilience. Two of the user stories (13%) did not support resilience, and instead described basic functional needs of the technology.

Table 3 provides an overview of which resilience factors the different user stories were mapped to. One should note that each of the five major resilience factors was supported by at least one user story. Further, one should note that a plurality of user stories (*n* = 5, 33%) was mapped to more than one resilience factor. The following are illustrative examples of user stories from the pilot TTX for each major resiliency factor:

- Shared Demand and Deviation Awareness (SDA, n = 3, 20%): As a physician, I want to be notified when the underlying models are changed, by whom, and why, so I can calibrate my trust in [The Technology];
- Response Coordination (REC, n = 1, 7%): As a physician, I want to provide organized and systemic notifications to patients when their qualifications for OB telemedicine change, so I can [unstated];
- Guided Local Control (GLC, n = 5, 33%): As a physician, I want to use local codes/rationales to override [The Technology] when necessary, so I can have appropriate localization and flexibility of data for my care team;
- Progressive Responding (PRG, n = 3, 20%): As a CMIO, I want to schedule planned periodic reviews of data and underlying models/criteria, so I can reevaluate current models;

- Maneuver Capacity (MAC, n = 2, 13%): As a CMIO (or Clinical Decision Support Team, CDST), I want to know when outliers occur or outcomes deviate from expectations, so I can evaluate the models and data earlier than periodic reviews;
- General (GEN, n = 4, 27%): As a physician, I want to use a standard set of questions to measure subjective experience, so I can [unstated].

Table 3. User story mappings.

User Story ID	SDA	REC	GLC	PRG	MAC	GEN
1				1		
2			1			
3						1
4						
5						
6						
7				1		1
8				1	1	1
9			1			
10	1		1			1
11	1		1			
12	1	1				
13						
14						
15			1		1	
Count	3 (20%)	1 (7%)	5 (33%)	3 (20%)	2 (13%)	4 (27%)

SDA = Shared Demand and Deviation Awareness, REC = Response Coordination, GLC = Guided Local Control, PRG = Progressive Responding, MAC = Maneuver Capacity, GEN = General.

3.2.2. Other Outcomes

After the primary goal of the TTX was to generate user stories that supported work system resilience, we found that it generated other outcomes that supported development efforts. A byproduct of conducting the process to extract user stories (described in Section 2.3.) is the enumeration of various stakeholders who played a role in the TTX. Prior to the RAD-XP, development efforts (e.g., requirements and designs) were focused primarily on the needs of physicians and patients. The TTX highlighted several other stakeholders and demonstrated their roles in supporting work system resilience. For example, the TTX showed how families of the patients would be valuable sources of information needed to populate the models, and how other entities such as the Chief Medical Information Officer (CMIO) and the Clinical Decision Support Team (CDST) would play key roles in measuring effectiveness of the technology and adapting it over time. Not only did the TTX result in the first user stories being generated for previously unrepresented users, but the enumeration of these additional users directly supports other user-centered design artifacts such as stakeholder maps [57] and user journey maps [58].

The TTX also generated several insights regarding the Concept of Operations (CONOPS) for the technology, i.e., how it would be used, administered, maintained, etc. These insights were items that were flagged by the RAD-XP team as having implications for the technology, but not being directly actionable by a developer. For example, we noted that the existence of user stories regarding the collection of subjective patient experience data (e.g., usability) implied that the technology would be regularly updated in response to such feedback. While such insights are not immediately implementable, they have implications for design efforts, since they show that the technology will need to be modular and adaptable based on the acknowledgment that the technology will change with regular periodicity.

3.2.3. Subjective Experience

As noted previously, subjective feedback from participants helped assess the TTX outputs in a meaningful context. Although we only received five responses to each survey, this sample represented 71% of all participants in the RAD-XP pilot (i.e., five of seven), which is a high response rate. Raw responses are presented in Table 4 for complete transparency; however, for the sake of simplicity, we discuss results here simply in terms of disagreement, neutrality, or agreement, irrespective of the magnitude (e.g., slightly agree, agree, and strongly agree are all considered agreement).

Table 4. Survey responses ($N = 5$).

Statement	**Agreement**						
	Strongly Disagree	**Disagree**	**Slightly Disagree**	**Neutral**	**Slightly Agree**	**Agree**	**Strongly Agree**
Survey 1: Post-TTX							
The TTX made me think about the technology from a unique or novel perspective.					1	4	
The TTX helped our team uncover and discuss assumptions underlying operational use of the technology.				1	1	3	
I actively participated and was engaged in the TTX.					2	2	1
I had fun and enjoyed participating in the TTX.						5	
With a guidebook, I think our team could design and execute future TTXs without external support.			1	1	1	2	
The TTX was an effective tool for generating user stories. [a]					2	3	
I think it would be valuable to conduct TTXs regularly throughout the entire development process. [a]				1	1	1	2
Survey 2: Post-Delivery of User Stories and Outputs							
I was satisfied with the quantity of user stories generated by the TTX.					2	3	
I was satisfied with the novelty of the user stories generated by the TTX.					2	3	
For the same time and effort as the TTX, I could have generated a higher quantity of user stories using a different method.	1	2	2				
For the same time and effort as the TTX, I could have generated higher novelty user stories using a different method.	1	1	3				
I was satisfied with the other outputs of the TTX (i.e., non-requirement outputs such as design seeds, assumptions, info for CONOPS, etc.).				1	1	3	
The TTX was an effective tool for generating user stories. [a]					2		3
I think it would be valuable to conduct TTXs regularly throughout the entire development process. [a]					2	1	2

[a] Statement was included in both the first and second surveys.

Table 4 provides an overview of the participant responses to Likert items from both surveys. In the first survey (immediately following the TTX) there was unanimous agreement that the TTX forced them to think about their technology from a novel perspective.

All but one participant agreed that the TTX helped them uncover assumptions they had made about how the technology might be used. Further, there was unanimous consensus that participants had fun and were engaged in the process, which is important to overcome challenges with stakeholder engagement in requirements elicitation [18,45]. Of particular interest is that only three of the five participants felt they could conduct their own TTX if given resources (instructions, templates, etc.). This shows that there is more work to be done to ensure that RAD-XP is an accessible tool that does not require expert design and facilitation of TTXs.

In the second survey (immediately following the delivery and briefing of outputs), there was unanimous satisfaction in the quantity and novelty of the user stories generated from the TTX, which aligned with unanimous disagreement from participants that they could have achieved more using other methods requiring a comparable level of effort to the TTX. Novelty was defined as the uniqueness of the user story, or the degree to which the user story would not have been elicited using another traditional elicitation method (e.g., stakeholder interviews or focus groups). These results helped contextualize the outputs of the TTX, showing that participants found them to be valuable. The only statement without unanimous agreement or disagreement was regarding satisfaction with the other outputs of the TTX (e.g., stakeholder analysis and CONOPS insights), where a single participant provided a neutral response.

Two questions were asked during both surveys. There was unanimous agreement from participants that the TTX was an effective tool for generating user stories, and that they would find it valuable to regularly use RAD-XP to conduct TTXs throughout the development lifecycle. Of particular interest is that the magnitude of agreement in both questions shifted positively between surveys. That is, participant views on the efficacy of RAD-XP only increased after seeing the outputs of the TTX.

In addition to Likert items, both surveys included open response questions aimed at understanding participant perceptions on the value of the TTX, and on how to improve it based on lessons learned. Given the small sample, no clear themes emerged from responses. Table 5 shows the open response questions from each survey, and an exemplar response that had the most actionable impact from the perspective of the RAD-XP team.

Table 5. Open response questions ($N = 5$) and example answers.

Statement	Example Response
Survey 1: Post-TTX	
What changes would need to be made to the TTX process, and/or what resources would you need to be able to conduct TTXs in the future without external support?	I think it would have been a bit better to have a clear separation on who were the developers versus end users. It would also have been good to have more of a setup of the technology in the beginning, including purpose, current stage in development, and current capabilities.
What were the benefits from conducting this TTX? Anything expected or unexpected?	It was interesting to hear how [end users] would deal with integrating [the tool] with other systems, and how they would make updates to the rules/algorithms being used.
Additional comments (general thoughts or pertaining to specific statements)?	I think part of the guidebook should talk about how the TTX should evolve when you're progressing through development of a technology.
Survey 2: Post-Delivery of User Stories and Outputs	
Do you have any additional thoughts about the relative costs and/or benefits of using the TTX to elicit software requirements to support work system resiliency?	For the time investment, I thought it was worth it. The outputs of the exercises may be helpful in future [customer] engagements.
How should this process be modified to support future TTXs? Why do you recommend these changes?	When training others, I think it would be great if you can do one initial session like you did with ours, then a second one where you have the developers/leads go through the process of creating a TTX session and carrying it out.
Additional comments (general thoughts or pertaining to specific statements)?	I plan to see how and where TTX and the TRUSTS framework could be looped into my various projects.

4. Discussion

4.1. Key Findings

The RAD-XP pilot allowed us to formatively assess the effectiveness of using a TTX to elicit requirements that support work system resilience. The following are key findings from the effort:

- The TTX was effective for generating user stories to support work system resilience. We found that 13 of 15 (87%) elicited user stories supported the resilience of the sociotechnical system into which the technology would be integrated;
- User stories generated from RAD-XP covered all of the big five factors of the TRUSTS framework;
- RAD-XP supported other user-centered design practices through the identification of stakeholders and various considerations for the operation, administration, and maintenance of the technology;
- Participants found the TTX to be an effective and efficient means to elicit user stories, aligning with previous findings from wargames and TTXs [21,24];
- Involvement of end users was crucial. Users drove discussion that resulted in user stories, and developers reported that such user commentary in the TTX was particularly helpful toward developing understanding of how the technology would be deployed in a sociotechnical system (see Table 5).

4.2. Limitations & Future Work

We have described an initial approach for applying TTXs to elicit user stories to increase the resiliency of sociotechnical systems. While successful, there were some limitations in the pilot study itself, and to the generalizability of the findings herein. A major limitation of the pilot study is that it was limited to aggregate subjective feedback on the outputs of the TTX. That is, participants provided responses to their general perceptions of the effectiveness and efficiency of the TTX as an RE method, but did not provide specific feedback on the quality, novelty, or utility of the individual user stories themselves. Further, the pilot was a single TTX applied on a single technology; it is unclear how these findings (e.g., the quantity or quality of resultant user stories) generalize to the development of different types of technologies.

Another limitation is the still emergent standardization of various aspects of RAD-XP. For example, the inject matrix artifact is a standardized way to generate injects by mapping scenarios to resilience factors in the TRUSTS framework; however, there was no analogous standardized method to write the associated discussion questions. These steps were completed notionally and somewhat subjectively, based on years of expertise in resilience engineering and TTXs, but will need to be further codified to make them repeatable and adaptable for future use cases where such expertise is not readily available.

To better triangulate findings, future work will involve conducting additional RAD-X-generated TTXs with different teams working on different types of technologies. We will collect more detailed feedback on individual user stories (quality, novelty, usefulness, etc.), and objectively assess their impact by tracking the proportion of user stories that were placed in the backlog, committed to a sprint, and/or implemented (a method described by Molich et al. [59]). Such an approach will allow us to better understand and verify the types of applications for which RAD-XP has the most impact. Following such research to refine RAD-XP, we are currently working to make the RAD-XP documentation and all associated artifacts (e.g., templates for the sensemaking interviews, scenario development, inject matrix, inject and prompt writing, MSEL, etc.) publicly available by the end of 2023.

Although analysis was a relatively short process, we may investigate the application of various technologies to automatically transcribe or process data from such events, which can be a bottleneck for analysis [18]. Some example technologies to consider may include using media such as video clips or screenshots for enhanced context and traceability in user stories [46], bots or conversational agents to help drive discussion in the absence of trained facilitators [47], or natural language processing tools to automatically highlight or extract user stories from transcribed conversations during TTXs [45,60,61].

Author Contributions: Conceptualization: S.L.D., T.F., A.L., E.B.; Formal Analysis: E.B., S.L.D.; Funding Acquisition: K.J.N., S.L.D.; Investigation: T.F., A.L., E.B., S.L.D., K.J.N.; Methodology: S.L.D., E.B., T.F., A.L.; Project Administration: K.J.N.; Writing—Original Draft: S.L.D., E.B., K.J.N.; Writing—Review & Editing: S.L.D., E.B. All authors have read and agreed to the published version of the manuscript.

Funding: This work was funded by the MITRE Independent Research and Development Program's Technology Futures Innovation Area. © 2023 The MITRE Corporation. All Rights Reserved. Approved for public release; distribution unlimited. Public Release Case Number 23-1929.

Institutional Review Board Statement: The study was conducted in accordance with the Declaration of Helsinki, and approved by the Institutional Review Board The MITRE Corporation (Protocol #2022-061, approved on 13 August 2022).

Informed Consent Statement: Informed consent was obtained from all subjects involved in the study.

Data Availability Statement: Data from this study are not available due to privacy restrictions as set forth by the institutional review board approving this study.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

References

1. Chopra, A.K.; Dalpiaz, F.; Aydemir, F.B.; Giorgini, P.; Mylopoulos, J.; Singh, M.P. Protos: Foundations for Engineering Innovative Sociotechnical Systems. In Proceedings of the 2014 IEEE 22nd International Requirements Engineering Conference (RE), Karlskrona, Sweden, 25–29 August 2014.
2. Aydemir, F.B.; Giorgini, P.; Mylopoulos, J.; Dalpiaz, F. Exploring Alternative Designs for Sociotechnical Systems. In Proceedings of the 2014 IEEE Eighth International Conference on Research Challenges in Information Science, Merrakech, Morocco, 28–30 May 2014.
3. Clegg, C.W. Sociotechnical Principles for System Design. *Appl. Ergon.* **2000**, *31*, 463–477. [CrossRef] [PubMed]
4. Roth, E.M.; Patterson, E.S.; Mumaw, R.J. Cognitive Engineering: Issues in User-Centered System Design. *Encycl. Softw. Eng.* **2002**, *1*, 163–179.
5. Woods, D.D. The Theory of Graceful Extensibility: Basic Rules that Govern Adaptive Systems. *Environ. Syst. Decis.* **2018**, *38*, 433–457. [CrossRef]
6. Dorton, S.L.; Harper, S.B. A Naturalistic Investigation of Trust, AI, and Intelligence Work. *J. Cogn. Eng. Decis. Mak.* **2022**, *16*, 222–236. [CrossRef]
7. Dorton, S.L.; Harper, S.B.; Neville, K.J. Adaptations to Trust Incidents with Artificial Intelligence. *Proc. Hum. Factors Ergon. Soc. Annu. Meet.* **2022**, *66*, 95–99. [CrossRef]
8. Dorton, S.L.; Ministero, L.M.; Alaybek, B.; Bryant, D.J. Foresight for Ethical AI. *Front. Artif. Intell.* **2023**, *6*, 1143907. [CrossRef] [PubMed]
9. Neville, K.J.; Pires, B.; Madhavan, P.; Booth, M.; Rosfjord, K.; Patterson, E.S. The TRUSTS Work System Resilience Framework: A Foundation for Resilience-Aware Development and Transition. *Proc. Hum. Factors Ergon. Soc. Annu. Meet.* **2022**, *66*, 2067–2071. [CrossRef]
10. Nemeth, C.; Wears, R.; Woods, D.; Hollnagel, E.; Cook, R. Minding the Gaps: Creating Resilience in Healthcare. In *Advances in Patient Safety: New Directions and Alternative Approaches*; Agency for Healthcare Research and Quality (US): Rockville, MD, USA, 2008; Volume 3.
11. Storm, B.L.; Schinnar, R.; Aberra, F.; Bilker, W.; Hennessy, S.; Leonard, C.E.; Pifer, E. Unintended Effects of a Computerized Physician Order Entry Nearly Hard-Stop Alert to Prevent a Drug Interaction: A Randomized Controlled Trial. *Arch. Intern. Med.* **2010**, *170*, 1578–1583. [CrossRef]
12. Mirsa, S.; Kumar, V.; Kumar, U. Goal-Oriented or Scenario-Based Requirements Engineering Technique–What Should a Practitioner Select? In Proceedings of the Canadian Conference on Electrical and Computer Engineering, Saskatoon, SK, Canada, 1–4 May 2005; pp. 2288–2292.
13. Raunak, M.S.; Binkley, D. Agile and Other Trends in Software Engineering. In Proceedings of the 2017 IEEE 28th Annual Software Technology Conference (STC), Gaithersburg, Maryland, USA, 25–28 September 2017.
14. Fowler, M.; Highsmith, J. The Manifesto for Agile Software Development. 2001. Available online: agilemanifesto.org (accessed on 1 July 2023).
15. Batra, M.; Bhatnagar, A. A Research Study on Critical Challenges in Agile Requirements Engineering. *Int. Res. J. Eng. Technol.* **2019**, *6*, 1214–1219.

16. Masood, Z.; Hoda, R.; Blincoe, K. Real World Scrum: A Grounded Theory of Variations in Practice. *IEEE Trans. Softw. Eng.* **2020**, *48*, 1579–1591. [CrossRef]
17. Ananjeva, A.; Persson, J.S.; Brunn, A. Integrating UX Work with Agile Development Through User Stories: An Action Research Study in a Small Software Company. *J. Syst. Softw.* **2020**, *170*, 110785. [CrossRef]
18. Kelly, S. Towards an Evolutionary Framework for Agile Requirements Elicitation. In Proceedings of the 2nd ACM SIGCHI Symposium on Engineering Interactive Computing Systems, Berlin, Germany, 19–23 June 2010; pp. 349–352.
19. Rodeghero, P.; Jiang, S.; Armaly, A.; McMillan, C. Detecting User Story Information in Developer-Client Conversations to Generate Extractive Summaries. In Proceedings of the 2017 IEEE/ACM 39th International Conference on Software Engineering (ICSE), Buenos Aires, Argentina, 20–28 May 2017; pp. 49–59.
20. Younas, M.; Jawawi, D.N.A.; Ghani, I.; Kazmi, R. Non-Functional Requirements Elicitation Guideline for Agile Methods. *JTEC* **2017**, *9*, 49–59.
21. Dorton, S.L.; Fersch, T.; Barrett, E.; Langone, A.; Seip, M.; Bilsborough, S.; Hudson, C.B.; Ward, P.; Neville, K.J. The Value of Wargames and Tabletop Exercises as Naturalistic Tools. In Proceedings of the HFES 67th International Annual Meeting, Washington DC, USA, 23–27 October 2023. article in press.
22. Caffrey, M. Toward a History-Based Doctrine for Wargaming. *Aerosp. Power J.* **2000**, *14*, 33–56.
23. Dorton, S.; Tupper, S.; Maryeski, L. Going Digital: Consequences of Increasing Resolution of a Wargaming Tool for Knowledge Elicitation. *Proc. Hum. Factors Ergon. Soc. Annu. Meet.* **2017**, *61*, 2037–2041. [CrossRef]
24. De Rosa, F.; Strode, C. Exploring Trust in Unmanned Systems with the Maritime Unmanned System Trust Game. *Hum. Factors Syst. Interact.* **2022**, *52*, 9–16.
25. De Rosa, F.; Strode, C. Games to Support Disruptive Technology Adoption: The MUST Game Use Case. *IJSG* **2022**, *9*, 93–114.
26. Dorton, S.L.; Maryeski, L.R.; Ogren, L.; Dykens, I.T.; Main, A. A Wargame-Augmented Knowledge Elicitation Method for the Agile Development of Novel Systems. *Systems* **2020**, *8*, 27. [CrossRef]
27. Smith, S.; Raio, S.; Erbacher, R.; Weisman, M.; Parker, T.; Ellis, J. *Quantitative Measurement of Cyber Resilience: A Tabletop Exercise*; DEVCOM Army Research Laboratory (US): Adelphi, MD, USA, 2022.
28. Deal, S.V.; Hoffman, R.R. The Practitioner's Cycles, Part 1: Actual World Problems. *IEEE Intell. Syst.* **2010**, *25*, 4–9.
29. Hoffman, R.R.; Deal, S.V.; Potter, S.; Roth, E.M. The Practitioner's Cycles, Part 2: Solving Envisioned World Problems. *IEEE Intell. Syst.* **2010**, *25*, 6–11. [CrossRef]
30. Bryant, D.J.; Martin, L.B.; Bandali, F.; Rehak, L.; Vokac, R.; Lamoureux, T. Development and Evaluation of an Intuitive Operational Planning Process. *J. Cogn. Eng. Decis. Mak.* **2007**, *1*, 434–460. [CrossRef]
31. Lombriser, P.; Dalpiaz, F.; Lucassen, G.; Brinkkemper, S. Gamified Requirements Engineering: Model and Experimentation. In Proceedings of the Requirements Engineering: Foundation for Software Quality, Gothenburg, Sweden, 14–17 March 2016; pp. 171–187.
32. Hahn, H.A. What Can You Learn About Systems Engineering by Building a Lego^{TM} Car? *INCOSE Int. Symp.* **2018**, *28*, 158–172. [CrossRef]
33. Dykens, I.T.; Wetzel, A.; Dorton, S.L.; Batchelor, E. Towards a Unified Model of Gamification and Motivation. In Proceedings of the Adaptive Instructional Systems. Design and Evaluation, Virtual Event, 24–29 July 2021; pp. 53–70.
34. Sutcliffe, A. Scenario-Based Requirements Engineering. *J. Light. Technol.* **2003**. [CrossRef]
35. Sutcliffe, R.B.; Ryan, M. Experience with SCRAM, a Scenario Requirements Analysis Method. In Proceedings of the IEEE International Symposium on Requirements Engineering: RE '98, Colorado Springs, CO, USA, 10 April 1998; pp. 164–171.
36. Van Lamsweerde, A.; Letier, E. Handling Obstacles in Goal-Oriented Requirements Engineering. *IEEE Trans. Softw. Eng.* **2000**, *26*, 978–1005. [CrossRef]
37. Lutz, R.; Patterson-Hine, A.; Nelson, S.; Frost, C.R.; Tal, D.; Harris, R. Using Obstacle Analysis to Identify Contingency Requirements on an Unpiloted Aerial Vehicle. *Requir. Eng.* **2007**, *12*, 41–54. [CrossRef]
38. Ponsard, C.; Darimont, R. Formalizing Security and Safety Requirements by Mapping Attack-Fault Trees on Obstacle Models with Constraint Programming Semantics. In Proceedings of the 2020 IEEE FORMREQ, Zurich, Switzerland, 31 August–4 September 2020; pp. 8–13.
39. Karolita, D.; McIntosh, J.; Kanji, T.; Grundy, J.; Obie, H.O. Use of Personas in Requirements Engineering: A Systematic Mapping Study. *Inf. Softw. Technol.* **2023**, *162*, 107264. [CrossRef]
40. Ferraris, D.; Fernandez-Gago, C. TrUStAPIS: A Trust Requirements Elicitation Method for IoT. *Int. J. Inf. Secur.* **2020**, *19*, 111–127. [CrossRef]
41. Lim, S.; Henriksson, A.; Zdravkovic, J. Data-Driven Requirements Elicitation: A Systematic Literature Review. *SN Comput. Sci.* **2021**, *2*, 1–35. [CrossRef]
42. Ramesh, B.; Cao, L.; Mohan, K.; Xu, P. Can Distributed Software Development be Agile? *Commun. ACM* **2006**, *49*, 41. [CrossRef]
43. Tanzman, E.A.; Nieves, L.A. *Analysis of the Argonne Distance Tabletop Exercise Method.*; No. ANL/DIS/RP-60983; Argonne National Lab. (ANL): Argonne, IL, USA, 2008.
44. Hansen, K.; Pounds, L. Beyond the Hit-and-Run Tabletop Exercise. *J. Bus. Contin. Emerg. Plan.* **2009**, *3*, 137–144.
45. Raharjana, I.K.; Siahaan, D.; Fatichah, C. User Story Extraction from Online News for Software Requirements Elicitation: A Conceptual Model. In Proceedings of the 2019 16th International Joint Conference on Computer Science and Software Engineering (JCSSE), Chonburi, Thailand, 10–12 July 2019.
46. Gall, M.; Berenbach, B. Towards a Framework for Real Time Requirements Elicitation. In Proceedings of the 2006 First International Workshop on Multimedia Requirements Engineering (MERE'06–RE'06 Workshop), Minnneapolis, MN, USA, 12 September 2006.

47. Rietz, T.; Maedche, A. LadderBot: A Requirements Self-Elicitation System. In Proceedings of the 2019 IEEE 27th International Requirements Engineering Conference (RE), Jeju, Korea, 23–27 September 2019; pp. 357–362.
48. Rasheed, A.; Zafar, B.; Shehryar, T.; Aslam, N.A.; Sajid, M.; Ali, N.; Dar, S.H.; Khalid, S. Requirement Engineering Challenges in Agile Software Development. *Math. Probl. Eng.* **2021**, *2021*, 6696695. [CrossRef]
49. Heyn, H.-M.; Knauss, E.; Muhammad, A.P.; Eriksson, O.; Linder, J.; Subbiah, P.; Pradhan, S.K.; Tungal, S. Requirement Engineering Challenges for AI-Intense Systems Development. In Proceedings of the 2021 IEEE/ACM 1st Workshop on AI Engineering–Software Engineering for AI (WAIN), Madrid, Spain, 30–31 May 2021; pp. 89–96.
50. Hosseini, M.; Shahri, A.; Phalp, K.; Taylor, J.; Ali, R.; Dalpiaz, F. Configuring Crowdsourcing for Requirements Elicitation. In Proceedings of the 2015 IEEE 9th International Conference on Research Challenges in Information Science (RCIS), Athens, Greece, 13–15 May 2015.
51. Dorton, S.L.; Maryeski, L.M.; Costello, R.P.; Abrecht, B.R. A Case for User Centered Design Satellite Command and Control. *Aerospace* **2021**, *8*, 303. [CrossRef]
52. Dorton, S.L.; Ganey, H.C.N.; Mintman, E.; Mittu, R.; Smith, M.A.B.; Winters, J. Human-Centered Alphabet Soup: Approaches to Systems Development from Related Disciplines. In Proceedings of the 2021 International Annual Meeting of the Human Factors and Ergonomics Society, Baltimore, MD, USA, 4–7 October 2021; Volume 65, pp. 1167–1170.
53. Klein, G. *Snapshots of the Mind*; MIT Press: Cambridge, MA, USA, 2022.
54. Dalpiaz, F.; Brinkkemper, S. Agile Requirements Engineering: From User Stories to Software Architectures. In Proceedings of the 2021 IEEE 29th International Requirements Engineering Conference (RE), Notre Dame, IN, USA, 20–24 September 2021.
55. Lucassen, G.; Dalpiaz, F.; van der Werf, J.M.E.M.; Brinkkemper, S. Forging High-Quality User Stories: Towards a Discipline for Agile Requirements. In Proceedings of the 2015 IEEE 23rd International Requirements Engineering Conference (RE), Ottawa, ON, Canada, 24–28 August 2015; pp. 126–135.
56. Lucassen, G.; Dalpiaz, F.; van der Werf, J.M.E.M.; Brinkkemper, S. The Use and Effectiveness of User Stories in Practice. In Proceedings of the Requirements Engineering: Foundation for Software Quality, Gothenburg, Sweden, 14–17 March 2016; pp. 205–222.
57. Walker, D.H.T.; Bourne, L.M.; Shelley, A. Influence, Stakeholder Mapping and Visualization. *Constr. Manag. Econ.* **2008**, *26*, 645–658. [CrossRef]
58. Marquez, J.J.; Downey, A.; Clement, R. Walking a Mile in the User's Shoes: Customer Journey Mapping as a Method to Understanding the User Experience. *Internet Ref. Serv. Q.* **2015**, *20*, 135–150. [CrossRef]
59. Molich, R.; Jeffries, R.; Dumas, J.S. Making Usability Recommendations Useful and Usable. *J. Usability Stud.* **2007**, *2*, 162–179.
60. Jindal, R.; Malhotra, R.; Jain, A.; Bansal, A. Mining Non-Functional Requirements Using Machine Learning Techniques. *e-Informatical Softw. Eng. J.* **2021**, *15*, 85–114. [CrossRef]
61. Franch, X.; Gómez, C.; Jedlitschka, A.; López, L.; Martínez-Fernández, S.; Oriol, M.; Partanen, J. Data-Driven Elicitation, Assessment and Documentation of Quality Requirements in Agile Software Development. In Proceedings of the Advanced Information Systems Engineering, Tallinn, Estonia, 11–15 June 2018; pp. 587–602.