New Frontiers for Requirements Engineering

David Callele
Department of Computer Science
University of Saskatchewan
Saskatoon, Canada
callele@cs.usask.ca

Krzysztof Wnuk
Department of Computer Science
Blekinge Institute of Technology
Karlskrona, Sweden
krzysztof.wnuk@bth.se

Birgit Penzenstadler
Department of Computer Science
California State University
Long Beach, CA, USA
birgit.penzenstadler@csulb.edu

***Abstract*— Requirements Engineering (RE) has grown from its humble beginnings to embrace a wide variety of techniques, drawn from many disciplines, and the diversity of tasks currently performed under the label of RE has grown beyond that encompassed by software development. We briefly review how RE has evolved and observe that RE is now a collection of best practices for pragmatic, outcome-focused critical thinking – applicable to any domain. We discuss an alternative perspective on, and description of, the discipline of RE and advocate for the evolution of RE toward a discipline that supports the application of RE practice to any domain. We call upon RE practitioners to proactively engage in alternative domains and call upon researchers that adopt practices from other domains to actively engage with their inspiring domains. For both, we ask that they report upon their experience so that we can continue to expand RE frontiers.**

***Index Terms*— Requirements engineering, cross-discipline, review, frontiers**

I. INTRODUCTION

RE can trace much of its implicit, and in some cases explicit, early lineage to techniques and methodologies developed for managing large-scale classic engineering projects such as bridges, dams and buildings (we use the term classic engineering as a synonym for the practice of what the general public perceives as engineering, generally large-scale physical constructs). Engineering project management, from which the basic concepts of formalized requirements and specifications evolved (*e.g.* in early forms of architectural drawings), was a relatively active research discipline until the late 1970s. Demand for large-scale software project management rose dramatically as we began the transition to the "information society" and many domain researchers turned their attention to applying engineering practice, principles and techniques to managing software development. As software systems became more complex, the Software Engineering discipline evolved to meet an increasing demand for methods that delivered upon the engineering principles of planning, predictability, and repeatability.

Early software project management was rife with failures as traditional engineering project management techniques were applied to the new domain. Royce [27] captured the then state of the art (for less successful projects) in what has become widely known as the waterfall model, a model directly descended from the techniques developed for large civil engineering projects. Waterfall delivered acceptable results for many projects in the physical world, where all stakeholders had the evidence of their senses to ensure that they had a shared understanding of project definition, scope and deliverables, but the relatively nebulous nature of software seemed to defeat many of the best intentions. Boehm's subsequent work on the cyclic nature of relatively successful projects [4], where numerous iterations across stages in the waterfall model occurred as understanding of the project evolved, is one of the clear antecedents of agile methodologies.

Practitioners and researchers began to realize that many of the problems could be traced back to deficiencies in what we now consider the domain of requirements engineering. In hindsight, the requirements were often incomplete, malformed, untestable or inconsistent and even if the requirements were accurate and well-formed, the stakeholders often had different understandings of what they meant. After all, requirements for physical entities focused more upon structure than behavior, the inverse of software artifacts, and adjustments were required. The complexity of applying engineering management principles to the development of a relatively intangible artifact having complex and dynamic structure and behavior, an artifact that could arbitrarily grow in complexity with relatively little increase in capital costs, was uncharted territory and many new challenges were identified.

Classic engineering is very much a practice of successive refinement, employed until the challenges are addressed with "known to be reliable" building blocks. This perspective can lead to debate as to whether there can truly be a practice of software engineering if the reliability of the software modules cannot be guaranteed. One can apply engineering practices to software development but the sheer complexity of the vast majority of systems means that the building blocks are not "reliable" in this sense and cannot, therefore, be considered "engineering".

The perceived lack of reliability and high degree of uncertainty has only been exacerbated with some software practitioners following "use and dispose" and "let the market test the software" business models. Both practices are anathema to classic engineers whose practice is often governed by legislation that assigns both civil and criminal liability for their professional activities if the result of their efforts fails in practice.

Software is also much more amenable to radical design [5][6], design that breaks with traditional methods to explore new frontiers. Radical design often has high initial failure rates until experience gradually transforms radical design into normal design. The low threshold for radical design in software-intensive systems (SIS) appears to contribute to a willingness to adopt radical designs. The associated risk may contribute to explaining why RE has its genesis within SIS and is more frequently applied in SIS than in other fields.

Into this morass of challenges and widely divergent perspectives stepped the practice of Requirements Engineering (RE). Initially an attempt to apply the best and most relevant engineering practices to software development, RE has significantly evolved beyond its roots. Today, RE practice draws from an incredible diversity of domains beyond engineering to establish its own perspective on best practices.

The power of requirements engineering for industrial projects lies in analyzing and decomposing complex problems into requirements and constraints that can be packaged into deliverables for project milestones. For software projects, neither the decomposition nor the packaging and delivery order are trivial tasks and heavily depend upon the nature of the product to be delivered and its operational context.

Although RE is inspired by many other domains and research fields, much of its application remains strongly focused on Software Intensive Systems (SIS) (products or services), those systems where software plays a significant or critical role.

In this work, we argue that RE is not just for SIS. Rather, we advocate that can also be characterized as pragmatic critical thinking, supported in the engineering tradition by a collection of best practices undergoing continual refinement for both general and specific application. We conclude that RE is a discipline that can be applied to very many (if not all) problem scenarios and that the advances in the RE discipline can and should be applied to other disciplines and problem frames beyond those experienced in SIS.

In support, we build upon prior work that reflects upon this diversity of origin (Alexander [1], Nuseibeh and Easterbrook [22]) and look at how special interest groups within the discipline have evolved as evidenced by the diversity of workshops associated with the RE conference series and comment upon their effects upon RE practice. We give a perspective on the diversity of tasks currently performed under the label of RE and how this diversity has grown beyond that encompassed by various textbooks, standards [19] and even the various "Book Of Knowledge" artifacts that might apply (*e.g.* SWEBOK [33], SEBOK [29], ISO 29148 [17], and the in-progress REBOK [26]). We propose an evolution of the working definition for, or description of, RE practice and advocate for greater interaction with other disciplines, particularly those from which we have drawn inspiration for our discipline. We conclude with a proposal for the evolution of RE toward a discipline with the deliberate intent of application to any domain. Of necessity this is a wide-ranging work and we apologize in advance for the brevity with which we treat certain topics and prior work.

II. What is Requirements Engineering?

One of the first authors' engineering professors once made the following (paraphrased) statement in class:

The historical record demonstrates that we learned everything that we need to know about engineering a very long time ago; consider what it took to build the Great Wall of China or the pyramids in Egypt and in many locations in South America. Since then, all we have done is become better at every aspect.

In the context of the class, the professor was helping the students to understand that we were standing on the shoulders of all who had gone before and that there have always been very intelligent people building things – people just as capable as we are today. Reflecting upon the centuries of engineering history and resulting artifacts we assert that the difference between then and now is in refinement, not substance; we still have to carefully consider what we are attempting, identify what we are going to do, identify the necessary resources, solve the challenges of design and implementation, and manage the whole process. The fundamental classroom message was that all of these major aspects were common practice thousands of years ago. In other words, critical thinking, a methodical approach, logistics and human resource management are fundamental elements of all successful engineering endeavors.

How did we transition from classic engineering to RE, a discipline that can be characterized (at least in part) as methodical critical thinking (leaving design, implementation and management to someone else)? The following is a brief overview of only some of the historical elements relevant to the current work; it is not intended to be a definitive history of RE.

In 1997, Ian Alexander [1] presented a personal reflection upon the evolution in development methodologies that led to the discipline of requirements engineering. In his work, he traced the evolution of the domain, from early perspectives of the computer as a scarce resource in the 1960s, through an engineering-oriented "precise description of components-to-be-built" in the 1970s that evolved into a systems perspective in the 1980s. RE eventually arrived at a more balanced "human and computer as a system" with the influence of disciplines such as psychology and ethnography in the 1990s. While not a traditional (formally peer-reviewed) research paper, his observations provide valuable insight into the intellectual history of our discipline.

There have been many definitions for RE presented over the years. We have chosen only a few representative examples to set the context for our discussion about what RE is and what RE does; this work is not meant to be an exhaustive review of these definitions.

Zave's [35] definition of RE is oft-cited and strongly justifies the founding focus of the RE discipline on SIS.

Requirements engineering is the branch of software engineering concerned with the real-world goals for, functions of, and constraints on software systems. It is also concerned with the relationship of these factors to precise specifications of software behavior, and to their evolution over time and across software families.

While appropriate in the context of the time (the author notes that the work was created at the time she was the chair of the Second IEEE International Symposium on Requirements Engineering), we demonstrate in this work that the RE discipline has, in the ensuing years, grown beyond the boundaries of SIS.

A few years later, Nuseibeh and Easterbrook [22] defined the core RE activities as elicitation, modeling and analysis, communication, agreeing (negotiation) and evolving (maintenance and management). They reflect upon the inherent SIS focus within the discipline and note that RE draws on many fields, including systems theory and systems engineering, cognitive psychology, anthropology, sociology and linguistics to name but a few. They also take note of the philosophic elements of RE including stakeholder beliefs (epistemology), observability (phenomenology) and objective truth (ontology). Their perspectives are remarkably congruent with those presented later in this work but, in their roadmap, tend to much more strongly focus upon specific RE challenges in an SIS context. Their final challenge, "Multidisciplinary training for requirements practitioners", is implied throughout our position in this work. Had the authors stated "Requirements engineering is discipline independent", we may not have felt compelled to author this work.

Cheng and Atlee [14] built upon [22] to investigate research directions in RE. They note that RE is difficult "because requirements reside primarily in the problem space whereas other software artifacts reside primarily in the solution space… Stated another way, requirements engineering is about precisely defining the problem that the software is to solve (*i.e.*, defining what the software is to do)…" If we relax their focus on software artifacts as solutions, their statement is congruent with our position presented here.

Returning to the historical perspective, in 2013 Mead [21] presented a history of the RE associated conferences. Woven within the history of the involved individuals are insights into how the research interests have continued to evolve over time. Mead also notes that "At times the conference has been criticized for being overly focused on software requirements rather than systems requirements engineering." and that "In spite of support from INCOSE, the focus on systems engineering has been sporadic." and that "Maintaining practitioner participation continues to be a challenge." We (respectfully) agree that RE may be overly focused on software requirements and are advocating that we broaden the RE practice domain.

As a counterpoint to typical perspectives as to what requirements *are*, Ralph [25] presents an alternative definition: "A requirement is a feature of a design object that is necessary to achieve a goal", a definition that is more closely aligned with our position. Ralph uses this definition to illustrate some of the ontological and epistemological challenges inherent in the way RE practitioners consider requirements. He posits that requirements can only exist at the intersection of design alternatives; if there is no intersection there can be no requirements, only features. This analysis leads him to ask whether (most, if not all) requirements would be more accurately considered as expressions of desire. This perspective, one arguably gathered "from a distance", is instructive for it shows that as we continue to improve our discipline, we can always benefit from a fresh perspective. This perspective, combined with that presented by Cheng and Atlee, influences our alternative definition for requirements engineering practice, presented later in this work.

A common refrain among RE researchers are the challenges with motivating practitioner adoption of RE research results [34]. Kaindl *et al.* [18] identified many of the challenges extant at the time of their work and it appears that most, if not all, of the challenges still exist today. We argue here that RE is a discipline that can be applied to any problem scenario. However, if the software industry doesn't widely adopt our practices, what is the hope for adoption beyond SIS as we advocate in this work? This challenge is the greatest threat to our work.

Sadraei, Aurum, Beydoun, and Paech [28] surveyed industrial RE practices within the Australian software industry and found a significant gap between theory (research results) and practice. Their methodology investigated only successful projects whose participants had high levels of domain knowledge, noting that even within these constraints there were non-trivial challenges with adoption and utilization of research results. Again, commitment to following RE practices is weak within industry, even within successful projects, a challenge that (unfortunately) our work addresses only as a by-product.

Of particular note and influence in our work is the adoption of viewpoint methods and ethnographic principles within RE [22][31]. Viewpoints and ethnographic principles introduced pressure to ensure that all stakeholders are identified and their input sought as a necessary element of the RE process [31]. From a pragmatic "market success" perspective, such as that advocated by Davis [15] and other prominent authors, they have been very important as mechanisms for ensuring communication in the language and cultural context of the listener – thereby helping RE practitioners to deliver what is intended.

We also informally canvassed a number of RE researchers and presenters at past RE conferences for their response to the question: "What is RE practice today?" A small sample of the (paraphrased) responses are presented here.

- RE is a pragmatic collection of best practices for the well-known tasks of elicitation, representation, prioritization, negotiation, triage, *etc.*
- RE is a form of ethnographic study where we study people (stakeholders) and we try to understand their needs.
- RE is the structured application of the situationally most appropriate techniques with appropriately scaled effort (requires evaluation of stakeholders and problem domain).
- The goal of an RE effort is to deliver a common understanding of the problem, factors to consider when choosing a solution, prioritizing the investments in developing a solution (without specifying how to solve the problem), capturing a solution representation that satisfies the needs of the community of interest.
- RE for SIS assumes a (problem, solution) tuple. However, goal modeling is about goals and problems rather than finding a sufficient system (solution) that can implement these goals so RE does not always need to generate a tuple – RE can be applied to problem definition only. RE practices could provide effec-

tive guidance for challenges in other domains; for example, a thorough problem investigation supporting mediation efforts between conflicting parties.

These responses have helped inform our definition for requirements engineering practice.

III. Another Perspective on Requirements Engineering

In Requirements Engineering: A Roadmap [22], Nuseibeh and Easterbrook make the following statements in their paper:

It has been argued that requirements engineering is a misnomer. Typical textbook definitions of engineering refer to the creation of cost-effective solutions to practical problems by applying scientific knowledge [Shaw90]. Therefore, the use of the term engineering in RE serves as a reminder that RE is an important part of an engineering process, being the part concerned with anchoring development activities to a real-world problem, so that the appropriateness and cost-effectiveness of the solution can then be analysed. It also refers to the idea that specifications themselves need to be engineered, and RE represents a series of engineering decisions that lead from recognition of a problem to be solved to a detailed specification of that problem.

We draw the reader's attention to their repeated reference to *engineering*; our interpretation is that this use of engineering is consistent with what we described earlier as classic engineering. Textbook definitions of engineering are very useful for educational purposes but they do not necessarily reflect the myriad aspects of practice. Because engineering textbooks are designed to be generic, their perspective usually does not include (local) societal constraints upon what engineers can and cannot do as part of their professional practice (as evidenced by legislation governing practice in numerous jurisdictions). For many people (and practitioners), engineering is an application of that part of science that is so well understood that the engineering product does not pose a threat to public safety. Finally, textbook definitions usually do not take into account the proliferation of titles with "engineering" in their label – whether or not those titles appropriately use the word engineering.

When taken within the larger, societal context, we suggest that a definition of engineering that more accurately represents actual usage is as follows.

Engineering is a body of practice that, when applied to the creation of a solution to a problem, ensures that the resulting solution artifacts will function as intended, with high probability, while meeting the constraints imposed by the problem stakeholders and the society within which the solution is initially deployed.

Following the same pragmatic approach, we propose the following definition for Requirements Engineering *practice* as it has evolved to the current state, including removing the SIS application domain constraint mentioned in some of the examples quoted earlier.

Requirements Engineering is a body of practice that, when applied to the identification and definition of a problem, its stakeholders and their constraints, improves the probability that the results accurately represent the problem and the solution constraints and that the results are presented in a manner that facilitates communication between stakeholders.

We draw the reader's attention in this case to the parallel mentions of probability. We know that perfection is likely impossible; these definitions make explicit the assertion that RE practice, when properly applied, improves the probability of success and is consistent with the definition used by the IREB [19]. Specific tasks, within our perspective on RE practice, such as specification and documentation are contained within "identification and definition". Verification and validation are contained within "accurate representation". This topic is discussed further in the next section.

If this a perspective on (or definition for) Requirements Engineering practice then what happened to the word *requirements*? Continuing with the keyword *constraints*, we suggest the following perspective.

A requirement is a stakeholder constraint upon the problem space and/or the solution space.

This alternate perspective identifies a problem space, a solution space, and the constraints upon the solution (the requirements). The traditional *must*, *should*, and *could* requirements constructs are all supported by this definition. It is important to note that the definitions for RE and for requirements do not require that RE be used to provide direction (what is to be done) toward a solution (how it is to be done). Instead, RE can be used to define just the problem space.

This definition for a requirement is (somewhat) mathematically formal. We do not necessarily advocate its use in general communication, particularly with stakeholders – the semantic overloading associated with *requirement* and *constraint* is sufficient to cause communications challenges if we attempt to redefine these words. The perspective might be more useful to RE researchers, providing other perspectives and motivations in areas such as modeling, verification and validation (particularly with constraint satisfaction techniques).

These definitions imply a shift toward even greater focus on the task of stakeholder identification [20], using their input to define the problem space then identifying the potential solution space. Proper application may require the participation of those tasked with creating the solution so that they can provide their input to the constraints upon the solution space. In this sense, our proposed perspectives upon RE practice and requirements also signals a shift toward the precepts advocated by Brown as *design thinking* [5] and utilized within RE practice, for example, by Pasman and Wieringa [24]. Within SIS, this is consistent with the advice given by the first author in the videogame development domain [9] where it was noted that the participation of the development team could reduce the number of iterations required to converge upon a common understanding of the realizable (within solution domain constraints) requirements.

IV. RE For SIS and non-SIS Domains

Upon reflection we noted that our focus was first upon stakeholder identification (assuming responsibilities typically associated with business analysts or marketing specialists) then working with stakeholders to capture their expressions of their

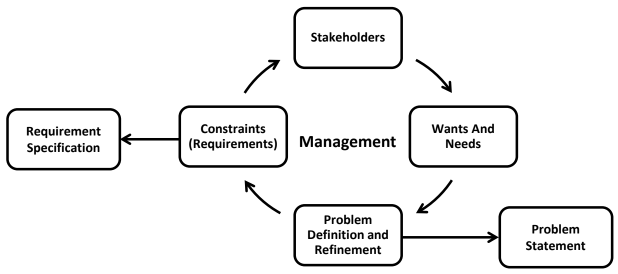
Stakeholders
Wants And Needs
Problem Definition and Refinement
Problem Statement
Constraints (Requirements)
Requirement Specification
Management

Figure 1. Generic RE Practice.

wants and needs. We continue to work with the stakeholders to capture and refine the problem definition, using them to verify that the problem statement was an acceptable representation. If we were addressing a scenario that required a solution (and not just problem identification) we would continue onward to identify the requirements and the constraints in the traditional manner. Given the expanded definition and accompanying discussion, is there a significant impact upon existing RE practices and processes? We do not believe that there is a significant impact and posit that a more general model should be considered. This model should be inclusive of existing RE tasks and practices, allow for ready adoption of new practices, and support the application of RE to non-SIS domains.

Following this path, we propose the following rearrangement of the well-known RE tasks, as shown in Figure 1, abstracting the discipline from its focus on SIS [17]:

1. **Stakeholders**: Identification and management, diversity, communications challenges, artifacts in the language of the stakeholder and of the domain, mediation
2. **Wants and Needs**: Elicitation, innovation and creativity, human factors, separating "what would be nice" from "what must occur"
3. **Problem Definition and Refinement**: Elicitation, Capture, Analysis, Definition (functional and non-functional, including quality characteristics and constraints), Representation (Natural language, Formalisms, Modeling, Rich-text, Visualization, etc.)
4. **Constraints (Requirements)**: Elicitation, Capture, Analysis, Definition, Representation, Prioritization and Negotiation, Triage, Scope management
5. **Management**: Measurement, Testing, Traceability and Accountability, Change Management, Productivity, Processes

The RE process is depicted as a cyclic iteration (sub-cycles are not shown for clarity) that proceeds (is managed) until the goal(s) for the process, such as a problem statement or requirements specification, are achieved.

Readers may note the absence of tools in this model. We well recognize that tools are always needed in support of all of the identified activities. However, we do not investigate the contributions of tools within this work.

Stakeholders is about the people (and/or their surrogates) affected by the problem, identifying them and building a community with shared perspective and a common language, managing their diversity and the inevitable conflicts. A taxonomy of stakeholders and a review of other stakeholder models is presented by Alexander [2].

Wants and Needs is about elicitation, exploration, creativity and innovation and (when constraints are applied) being pragmatic about identifying the core needs, including goals and objectives.

The Problem is about the practical aspects of wants and needs, methodically eliciting, capturing, analyzing and representing the stakeholder expressions and, when appropriate, leading to the later definition of the functional and non-functional requirements.

Constraints (Requirements) is about the necessary constraints upon the problem space and upon the solution space, including how the solution might be pursued (constraints such as budget, personnel, task-ordering, *etc.* are also considered). Our recommendation to include those responsible for delivery during the negotiation, prioritization and triage steps (carefully managed to ensure that possible requirements are not prematurely rejected due to implementation concerns) is expected to lead to an increased number of iterations in the requirements process. While this can make the requirements process more expensive, we posit that it is likely less expensive than completing a requirements process only to discover an unexpected implementation constraint and restarting the RE process. Including the delivery team during the RE process may also lead to improved recognition of the value delivered by RE.

Management is about making the RE process work as needed, ensuring that the result is as close as possible to what was intended.

For each of these five aspects, the relative investments in their execution can vary. For example, if the problem and/or the constraints have been clearly identified by others, then the RE practitioners need only validate the results of the prior work. Similarly, if there is little difference between the individual stakeholders' wants and needs, prioritization, negotiation and triage are relatively simple tasks compared to a high conflict situation. Finally, when the RE practitioner is using RE techniques solely for the purpose of identifying the problem (as in the example of eliciting contextual information for a mediation scenario in Section V), proceeding deeply into requirement and constraint identification may not be necessary.

For many years, development teams have delivered artifacts that met the requirements that they were given. Unfortunately, all too often those requirements had many issues – they were poorly formed, inaccurate, etc.; they were simply not the proper requirements for the system. There are many reports [9] about project problems and failures that, upon reflection, can be traced back to issues with requirements: The development team did things right, they just did not do the right thing.

RE practitioners might be tempted to make the claim that these mismatches were due to inadequate RE processes but whose "fault" is that? The consequence has been undeniable: The lack of faith in "formal" requirements, given the highly visible nature of some past requirements (and project) failures, has led to credibility and adoption challenges for RE. After all, requirements challenges are oft-cited as one of the justifications for the Agile movement; practitioners were tired of coding the wrong feature, for a problem that did not exist, that came from a misrepresented stakeholder. Agile proponents do not dismiss the need for requirements, instead they rely upon existing shared understanding to reduce the need for formal requirements [11]. Unfortunately, many software developers who believe that they are following agile practices are simply performing exploratory development (not agile development). Sometimes exploratory development works well but, in our personal experience, "coding to capture requirements" is far more expensive than an RE effort led by a competent practitioner. We posit that this trend has also contributed to adoption challenges for RE practices.

If we are advocating generalization of RE practice, we should reflect upon whether the elements of RE practice are uniquely applicable to SIS. One can posit that SIS are unique engineering artifacts in that they can create purely abstract artifacts. While some software has attributes that can be directly sensed by the user via the interface, there can be significant elements that are not directly perceived by the user. In that sense there is a (we feel, weak) argument to be made that RE practice is unique to abstract artifacts. However, if one considers the consequences and side-effects of the 'hidden' software then this argument is relatively meaningless. And, as evidenced by the following workshops discussion, RE has been successfully applied to domains as diverse as the law, corporate governance, creativity, entertainment and sustainability.

V. Crossing Domains

From the beginning of the discipline, Requirements Engineering was characterized by two distinct phenomena: (1) a recognition of the need for openness to and inspiration from other disciplines and (2) the resulting diversity of interests and expertise areas within requirements engineering research. This focus on diversity and openness resulted in inspiration from system engineering [3] and problem decomposition principles drawn from classic engineering. As the discipline evolved [32], particularly when software aspects began to dominate system development and maintenance costs, the need to draw inspiration from other domains became stronger and more urgent. As the projects and products continued to expand in size and complexity (sometimes with operational lifespans measured in decades), the need to study the sociological and human aspects of system development became increasingly important.

These trends helped RE better understand how to support the early phases of system development (problem identification and definition, decomposition and specification), ongoing support and maintenance, system evolution and system phase-out or decommissioning. It seems obvious that these skills could be applied to any domain that faces complex problems requiring non-trivial analysis. Therefore, it is somewhat surprising that the advances, experiences and techniques developed with the help of various other domains are not returned back to these domains, complete with a thorough discussion of the possible usage scenarios. An example is Steve Easterbrook's climate change all actors initiative [15] wherein RE methods are used in a methodical and pragmatic manner to deeply investigate the issue. He notes, in particular, that our community can contribute via our skillsets in these areas (among others):

- *understand and model complex inter-related systems and systems-of-systems;*
- *analyze and prioritize multi-stakeholder requirements;*
- *build useful abstractions and problem decompositions;*
- *manage and coordinate large-scale open source design communities;*
- *study and understand evolutionary forces at play in technical infrastructures;*
- *identify, diagnose and repair bugs in socio-technical systems;*

As a result of this work, those interested in the domain were provided with a structured map of challenges within the domain. The RE result in this case was not requirements for a software system but an improved understanding of a concept that is intangible and which can easily grow in complexity. Therefore, we believe that one of RE's significant benefits is that it can deliver an improved conceptual understanding of aspects of the domain under investigation. Even if the domain grows, RE's use of structured decomposition allows practitioners to address the changes in (relative) isolation from those aspects that have already been addressed, enhancing efficiency and providing opportunities for verification and validation at the boundaries between aspects.

The first author's earliest work published in this community [9] demonstrated how the analysis of post mortem project reports in the videogame production process could be used to improve the pre-production to production handoff. That work strove to identify an RE process that would capture the game designer's vision in a manner that the production team could use to guide the development process, with a reasonable expectation that the resulting software artifact would reflect the game designer's vision.

A videogame is a sophisticated software simulation whose attributes and capabilities matter very little to the customer, they are irrelevant to the customer as long as they do not fail – they exist only to support the creation of an experience for the customer. In other words, all of the work done by the technology team, the media teams and the game design team do not have intrinsic value to the user except for their byproduct – their ability to induce a user experience. An RE process focused only upon the software artifact would miss delivering the real value to the end customer – the playing experience. In [9], RE was used to capture the intended player experience *and* the requirements for the software needed to deliver that experience.

The authors have reported prior work using RE techniques in risk analysis [10] and intellectual property policy [11]. The first author's professional practice specializes in mediation and remediation of issues (usually failures and the resulting conflicts) in IT product and service contracts. In this instance, the full range of RE techniques are used to identify what has gone wrong, the source(s) of the fault(s) and probable solution paths. When a conflict scenario exists, RE techniques for elicitation, representation, prioritization, negotiation and triage have all been successfully employed to bring the situation "under control".

In the examples noted here and in the workshop examples below, we see that RE techniques were applied in widely divergent domains, delivering stakeholder identification, constraints, problem definition and system definition.

A. Workshops in RE

Throughout the history of the RE conferences the principal focus has been upon the pursuit of improving the requirements. However, there have always been those who have investigated the application of the techniques to other domains or adopted interesting concepts from other domains. These applications have usually focused upon SIS within those other domains (*e.g.* security, automotive) but some of them (discussed below) have ventured further afield. These workshops are, in essence, "expressions of interest" in RE for specific application domains, often with SIS applications but also for non-SIS applications.

MERE – Multimedia and Enjoyable Requirements Engineering (since 2006) – Oliver Creighton, Bernd Bruegge. This workshop series expanded the boundaries of what a requirement *is* to what a requirement *could be* – a mechanism for ensuring communication between stakeholders, a more human approach to requirements. The main contributions are openness to new mechanisms for requirements capture and representation, enhanced stakeholder communication and communicating in the language of the stakeholder.

RELaw – Requirements Engineering and Law (since 2007) – Annie Antón, Travis Breaux, Dimitris Karagiannis and John Mylopoulos. The workshop series has had a common theme of "the intersection of laws, policies and standards with software and system requirements to understand better ways to manage compliance, accountability and traceability" but the participants and topics have exhibited some of the greatest diversity within the RE conference workshop series. The strong correlations between the legal domain and the kind of analyses and structures used in RE practice is explored. The main contribution is the application of RE precision to one of our most critical societal constructs, using RE to assist legislators to improve the quality of their legislation.

RE4SuSy – Requirements Engineering for Sustainable Systems (since 2012, originally at REFSQ) – Birgit Penzenstadler, Martin Mahaux, Camille Salinesi. The workshop series focusses on investigating the use of RE techniques for sustainability. Noting that sustainability is one of the largest challenges facing society today, the workshop chairs advocate the use of RE practices to help develop high-quality solutions within the larger perspective of socio-technical systems. The main contribution is the application of RE to a societal level challenge, significantly increasing the breadth of possible domains to which RE could be beneficially applied.

UsARE – Usability and Accessibility focused Requirements Engineering (since 2012, originally at ICSE) – Tiziana Catarci, Anna Perini, Norbert Seyff, Shah Rukh Humayoun, Nauman A. Drawing heavily from realms such as human factors, HCI and psychology, UsARE focuses upon ensuring that traditional RE practices include usability and accessibility as first-order requirements, from the very beginning of a project, rather than relegating them to non-functional requirements or the dreaded "The application must be simple to use". The main contribution is the strong emphasis on human factors, HCI and psychology, enhanced stakeholder involvement, and user focus.

RE Interactive (since 2013), Creative Collisions – Martin Mahaux, David Callele. This initiative focused on promoting interactions wherein participants were randomly paired with other participants for a 15 minute working session where they mixed their knowledge and expertise to produce some creative result: hence the term "collisions". Almost any kind of output was possible but the grand challenge was to create a research problem statement and research program mission statement in just 15 minutes. Participants were skeptical but the final results amazed everyone who was there. The most common comment (paraphrased here): "If only the rest of my life was as productive as these 15 minutes have been!" The main contribution is the gamification of RE brainstorming, demonstrating that enforced interaction with someone you did not know, with clear deliverables and a clear time limit, can produce unexpected results.

Summary – These workshops and initiatives are the first steps toward bridging domains and facilitating an exchange between disciplines. However, each of the workshops has, at times, struggled with consistently attracting presenters and participants from outside the RE community. More thought and initiative needs to go into actually crossing those bridges in both directions. A possible solution for greater participation from outside the RE community is to proactively engage with a researcher in the other domain and to conduct research that will deliver valuable results in each domain.

B. Competitors and Collaborators

Requirements Engineering is a collaborative discipline focusing upon ensuring that stakeholder needs are met, RE relies heavily upon effective communication. In this context, we reflect upon who are the natural collaborators with requirements engineering practitioners and who may perceive RE as competition if the discipline were to endorse our position.

The **Business Analyst** (BA) is often perceived as an input provider to RE practice. Some consider the BA role to be bespoke, unique to each project, very "traditional" in their approach and not "agile". This perspective can lead to communications challenges with the rest of the team if it runs counter to their philosophy. Independent of the BA's practices, a requirements engineer is expected to work with the results from the business analyst, work based, at least in part, upon market research that identifies (customer) stakeholder wants and needs. Sometimes, however, the requirements engineer might carry out the tasks of a BA if there is no such designated role [10]. As such, the BA may be concerned that RE practitioner may be encroaching upon their responsibilities and possibly rendering them redundant.

The **Product Manager** is responsible for product definition, overall project scope, scope management and communication with the development team. RE assumes that the product manager will make use of RE practice in support of their management role. Product Managers often work as a requirements engineers when defining product scope and requirements and are expected to be natural collaborators.

The **Project Manager** is responsible for product development and delivery within given resource constraints. Project management often doubles as a stakeholder representative and part-time requirements manager. Most often responsible for requirements prioritization, an expanded role for RE may reduce their responsibilities.

The **Systems Engineer** is responsible for ensuring that all elements within the proposed solution can work together and meet the functional and nonfunctional requirements. An expanded role for RE could reduce the oversight necessary from the systems engineer but it is unlikely that RE practitioners would replace the system engineer's technical competencies in a given domain.

The **Systems Architect**, in an SIS context, ensures that all elements within the software elements of the proposed solution can work together and meet the functional and nonfunctional requirements. As with the systems engineer, an expanded role for RE could reduce the oversight necessary from the systems architect.

The **Product Designer** is responsible for the human-centric elements of the proposed solution. Their goal is to ensure that the customer stakeholder experience is as positive as possible. As RE practitioners gain greater experience in the design space, the product designer role may focus more upon the "intangibles" of hedonic design elements and less upon the functional aspects [13].

These disciplines utilize a wide variety of methodologies such as design science, open innovation and even agile methodologies as a surrogate for parts of the RE process. Imagine what might be achieved by actively engaging with these practitioners!

We believe that both of these efforts, the building of bridges to other disciplines in the workshops and the dedicated seeking out of natural collaborators from other areas, have the potential for high returns on investment with relatively little risk. Expanding the frontiers of RE beyond the development of SIS can only make RE a more effective practice with the potential for delivering significant benefits to other disciplines.

VI. New Frontiers for RE

Throughout this work we have advocated for relaxing the apparent restriction of the requirements engineering domain focusing its practice on software intensive systems. Observing how the field has evolved, and our experiences as practitioners, we advocate that RE should continue to explore new frontiers beyond software intensive systems in an effort to expand the community and keep it vibrant. Requirements engineering could evolve into a discipline wherein the practitioner is a facilitator of a pragmatic, outcome focused, critical thinking process, applicable to any domain. To this end, we have discussed alternative perspectives on requirements engineering practice and on requirements themselves. These alternative perspectives support the expansion of RE beyond software intensive systems, using as fundamental concepts a problem space, a solution space, and the constraints upon both (the requirements).

At no time do we advocate that RE should not continue to be used for SIS or that the main focus of RE should be something other than SIS. However, we do advocate for expanding RE practice and research to embrace new frontiers.

We ask that **industry practitioners** observe whether or not they are applying their RE skills to domains outside of software intensive systems and, if they are, report back upon their experiences to our community. We further ask that **academic researchers** look toward the fields that they are drawing upon for inspiration and to attempt to present the results of their work not just to the RE community but also to those fields from which they drawn inspiration. In both cases, interaction with these new domains can only help to keep our discipline invigorated.

We perceive that, as a first step, we can **become even more stakeholder focused** than we are today. There is much that we can learn from business analysts, marketing specialists and design thinkers to expand our field of practice – perhaps even more so in the other direction [20]. As practitioners, we can also consider applying our skills to the task of problem definition (only) and not always assume that there must be a solution to the problem as part of our process.

Using a simple **SWOT analysis**, we believe that the fundamental **strength** of this approach is that we are basing our expanded practice upon a solid methodological base with over 25 years of research and practical results. The new opportunities will help keep our domain dynamic as we seek to expand our frontiers. Our **weaknesses** include a lack of adoption within SIS and this may impair our credibility. As practitioners, we may lack certain domain skills, such as those used in business case analysis, but these can be acquired in a relatively short time given the skill set upon which we are building.

The **opportunities** are many. Business analysts and marketing researchers exist for a reason and there is much that we can learn from them. We expect that the relative rigor of RE processes can also benefit them and help them to expand their domain of interest as well. Perhaps we will also be able to pull some of their practices into academic research and further enhance the credibility of both domains. As for **threats**, we may have some resistance from the business analyst community who may feel that we are encroaching upon their domain. As noted in the weaknesses, we also have a demonstrated a lack of acceptance even within the software development community and our relevance may be questioned.

RE could be positioned as a spectrum of techniques that are applied as appropriate to adequately meet stakeholders' desires and needs; from no formal RE efforts when there are easier or more convenient techniques through to the most rigid and formal of techniques. We have also advocated for participation in the requirements process by members of the development team and while this participation must be carefully managed to ensure that the requirements process is not unduly affected by implementation discussions, this participation may also result in greater acceptance of RE practice.

In summary, the evolution of RE can be guided by greater outreach into other disciplines, generating greater awareness of our body of knowledge, and with a deliberate effort to apply RE techniques to alternate domains. Success in both of these endeavors could allow RE to become recognized as a vigorous discipline with a greatly expanded community.

VII. Future Work

This study identifies numerous opportunities for RE research and practice.

First, a systematic review of prior work in RE that identifies methodologies and techniques drawn from other disciplines would provide a comprehensive reference for cross-discipline opportunities. A similar review could identify specific techniques and skill sets from other disciplines that RE practitioners could acquire to enhance their practice. In particular, identifying the skill set and the expected benefit would be of great value, especially if there are some "easy wins" that have escaped our notice to date.

Secondly, there are also opportunities within the scope of traditional RE. There are ongoing efforts to understand the interactions between RE and agile methodologies. Deeper investigations of the role of "shared understanding" within development teams and its interactions with traditional RE are needed. Measuring this shared understanding and its effects on the (required) intensity of the RE process will be particularly challenging. Deeper investigation of the potential benefits (*e.g.* return on investment, enhanced credibility for and adoption of RE) associated with including the delivery team in the RE process is needed.

Thirdly, a cost-benefit analysis of "coding to capture requirements" compared to RE efforts could help practitioners identify the appropriate investment for RE efforts. Further investigations of the effects of radical design could help RE identify an appropriate role within that context.

Fourthly, fundamental tenets of RE include the positions that investing in RE reduces risk and improves project outcomes. Are these tenets true? Is it possible to measure the scale of 'failures' in software development as the domain has evolved and normalize these failures to project complexity? What metrics would be appropriate? Is it possible to determine if we are making the same mistakes now as we did twenty years ago? Is the frequency of these mistakes the same or are we making progress?

Finally, we have proposed expanding the frontiers of RE to seek new opportunities for our community to make contributions to the human condition. What would RE, as an independent discipline, a discipline both applicable to SIS and arbitrary problem domains, look like? This paper brings supporting evidence that RE can not only support important societal constructs (laws and legislation) but also tackle important societal challenges (*e.g.* sustainability) whose resolution we are all dependent upon. Continued efforts are needed to further highlight the importance of RE and related disciplines to the health of our societies.

There are many more opportunities for this community to make contributions to the human condition. It is our sincere hope that the RE community sees fit to embrace these opportunities.

Acknowledgment

We would like to take this opportunity to thank all of the members of the RE community for their incredibly inspiring work and the warm welcome that they gave each of us as we joined the community. Please consider this work our tribute to all that you have accomplished, separately and together. Finally, a special thank you to the reviewers whose suggestions greatly improved this work.

References

[1] I. Alexander, "A Historical Perspective on Requirements", Requirenautics Quarterly, the Newsletter of the BCS RESG, October 1997, Found online at http://www.scenarioplus.org.uk/papers/historical/historical.htm

[2] I. Alexander, "A Taxonomy of Stakeholders, Human Roles in System Development." International Journal of Technology and Human Interaction, Vol. 1, pp. 23-59, 2005.

[3] Blanchard, Benjamin S., Wolter J. Fabrycky, and Walter J. Fabrycky. "Systems engineering and analysis." Vol. 4. New Jersey; Prentice Hall, 1990.

[4] B. W. Boehm, TRW Defense Systems Group, "A Spiral Model of Software Development and Enhancement." Found online at http://csse.usc.edu/TECHRPTS/1988/usccse88-500/usccse88-500.pdf

[5] W. Vincenti, "What Engineers Know and How They Know It." John Hopkins University Press, Baltimore and London. 1990.

[6] M. Jackson, "What Can We Expect From Program Verification." Computer, Vol. 39(10), 2006.

[7] T. Brown, "Design thinking." Harvard Business Review, 86(5), 2008, pp. 84-92.

[8] J. A. Bubenko, "Challenges in requirements engineering." Requirements Engineering, 1995, Proceedings of the Second IEEE International Symposium on. IEEE, 1995.

[9] D. Callele, E. Neufeld, and K. Schneider. "Requirements engineering and the creative process in the video game industry." Proceedings of the 13th IEEE International Requirements Engineering Conference (RE'05), IEEE, 2005, pp. 240-250.

[10] D. Callele, B. Penzenstadler, K. Wnuk, "Risk Identification at the Interface between Business Case and Requirements." In 19th International Working Conference on Requirements Engineering Foundation for Software Quality, April 2013, pp. 253-268.

[11] Glinz, M. & Fricker, S.A. "On shared understanding in software engineering: an essay" Computer Science – Research and Development (2015), Vol. 30(3) pp. 363-376.

[12] D. Callele and K. Wnuk. "More Than Requirements: Applying Requirements Engineering Techniques to the Challenge of Setting Corporate Intellectual Policy, an Experience Report." In Proceedings of the 4th International Workshop on Requirements Engineering and Law (RELAW '11). August 2011, Trento, Italy, pp.35-42.

[13] A. Sutcliffe, P. Rayson, C. Bull, P. Sawyer, "Discovering affect-laden requirements to achieve system acceptance." Proceedings of the 22nd IEEE International Requirements Engineering Conference (RE'14). IEEE, 2014. p. 173-182.

[14] B. H. C. Cheng and J. M. Atlee, "Research directions in requirements engineering," in 2007 Future of Software Engineering (FOSE '07). Washington, D.C., IEEE Computer Society, 2007, pp. 285-303.

[15] A. Davis, "Just enough requirements management: where software development meets marketing." Addison-Wesley, 2013.

[16] S. Easterbrook, "Climate change: a grand software challenge." Proceedings of the FSE/SDP workshop on Future of software engineering research. ACM, 2010.

[17] ISO/IEC/IEEE 29148:2011 Systems and software engineering – Life cycle processes – Requirements engineering http://www.iso.org/iso/catalogue_detail.htm?csnumber=45171

[18] H. Kaindl, S. Brinkkemper, J. Bubenko, Jr., B. Farbey, S. Greenspan, C. Heitmeyer, J. Leite, J. Mylopoulos, N. Mead, and J. Siddiqi, "Requirements engineering and technology transfer: obstacles, incentives and improvement agenda." Requirements Engineering Journal, Vol. 7(3), Springer-Verlag London Limited, 2002, pp. 113-123.

[19] M. Glinz, "A glossary of requirements engineering terminology." Standard Glossary of the Certified Professional for Requirements Engineering (CPRE) Studies and Exam, Version 1 (2011).

[20] K. Wnuk, "Involving relevant stakeholders into the decision process about architecturally significant assets: challenges and opportunities." Accepted for International Workshop on decision Making in Software ARCHitecture (MARCH) workshop. Collocated at ICSA 2017, April 4, 2017, Gothenburg, Sweden.

[21] N. R. Mead, "A history of the international requirements engineering conference (RE) RE@21." Proceedings of the 21st IEEE International Requirements Engineering Conference (RE'13). IEEE, 2013. Rio de Janeiro, 2013, pp. 21-28.

[22] B. Nuseibeh and S. Easterbrook, "Requirements engineering: a roadmap." Proceedings of the Conference on the Future of Software Engineering (ICSE '00). New York: ACM, 2000, pp. 35-46.

[23] T. O'Reilly, "What is Web 2.0: Design patterns and business models for the next generation of software." Communications & Strategies 1 (2007). MPRA paper 4578.

[24] G. Pasman and E. Wieringa, "Landing design thinking in industry: making software for bookkeeping, but not in a bookkeeping way." Proceedings of the 2011 Conference on Designing Pleasurable Products and Interfaces (DPPI '11), 2011, ACM, New York, NY, USA.

[25] P. Ralph, "The illusion of requirements in software development." Requirements Engineering, Vol. 18 (3), 2013, pp. 293–296.

[26] M. Aoyama, T. Nakatani, and S. Saito, "REBOK Manifest: Towards a Requirements Engineering Body of Knowledge." Proceedings of the 18th IEEE International Requirements Engineering Conference (RE'10). IEEE, 2010. Sydney, Australia 2010, pp. 383-384.

[27] W. Royce, "Managing the Development of Large Software Systems." Proceedings of IEEE WESCON 26 (August), pp.1-9.

[28] E. Sadraei, A. Aurum, G. Beydoun, and B. Paech, "A field study of the requirements engineering practice in Australian software industry." Requirements Engineering Journal, Vol. 12(3), Springer-Verlag London Limited, 2007. pp. 145-162.

[29] SEBOK Systems Engineering Body of Knowledge http://sebokwiki.org/wiki/Guide_to_the_Systems_Engineering_Body_of_Knowledge_(SEBoK)

[30] Shaw, M., "Prospects for an Engineering Discipline of Software." IEEE Software, 7(6), pp. 15-24.

[31] I. Sommerville, T. Rodden, P. Sawyer, R. Bentley, M. Twidale, "Integrating ethnography into the requirements engineering process." Proceedings of the IEEE International Symposium on Requirements Engineering, 1993. pp. 165-173.

[32] I. Sommerville, P. Sawyer, "Requirements engineering: a good practice guide." John Wiley & Sons, Inc., 1997.

[33] SWEBOK Software Engineering Body of Knowledge https://www.computer.org/web/swebok/

[34] R. Wieringa, P. van Eck and J. Mylopoulos, "Requirements engineering conferences: Wither industry tracks?" Requirements Engineering Conference (RE), 2013 21st IEEE International, Rio de Janeiro, 2013, pp. 349-352.

[35] P. Zave, "Classification of Research Efforts in Requirements Engineering." ACM Computing Surveys, 29(4): 315-321.