Broadening the Lens:

Toward the Collective Empathic Understanding of Product Requirements

Robert C. Fuller
Electrical and Computer Engineering
The University of British Columbia
Vancouver, Canada
e-mail: rfuller@ece.ubc.ca

***Abstract*— Many software product companies have embraced the agile paradigm and gone on to create cross-functional product development teams that fully own their product. The expectations of these teams are very different than of development teams in a disciplined software development environment. The study underway examines how these empowered cross-functional product teams, as a collective, create and nurture a shared mental model that accurately represents the external product domain and its realities and that provides the context for understanding the requirements. We also examine external factors that allow for these teams to develop these capabilities while less-empowered teams cannot. Using Constructivist Grounded Theory, we study individuals and teams in several companies and varied product domains. We find that certain organisational factors play a significant role and we also examine an essential dynamic of broadening the lens and blurring boundaries that cross-functional product teams employ in order to not only fully embrace product planning but also to grok the domain for their products.**

Keywords - empathy-driven development; collective sensemaking; design science; requirements validation; product team organisation.

I. INTRODUCTION

Product development is a social process; thus, the dimension of the organisational model and dynamics is the elephant in the room, a critical factor for success or failure of software products.

This study builds our earlier work [1] that studied software product teams that displayed varying degrees of collective grokking. In that study, we found that the organisational model surrounding the teams had a profound influence on whether the teams could grok the product requirements at all. Building upon that work, we use the Constructivist Grounded Theory method (described further in Section V) to examine characteristics of collective team grokking of the product domain and we also examine how the extra-team organisational model affects the team's ability to own increasingly comprehensive product planning.

We use the concept of broadening the lens as an explanatory mechanism that Cross-functional Product Teams (CFPTs) use to explore further and innovate more and we also look at some of the prerequisite conditions in order for teams to do this.

Grokking is cognitive empathy, coupled with skilled perspective-taking. We use a definition of cognitive empathy to be "*the ability to imaginatively step into another domain, understand the perspectives of those in that domain, and use that understanding to guide decisions"* [2,x]. Increasingly, the success of software product development teams depends on the degree to which the team *collectively* groks not only the product requirements themselves but also, and importantly, the *context* for those requirements.

The remainder of this paper is structured as follows: Section II – Background and Problem provides an overview of the historical background and description of the research problem. Section III – Research Motivation and Focus describes what we're aiming to achieve and a brief description of the research scope. Section IV – Related Work positions this study with respect to three related areas of research. Section V – Method and Status of the Research overview the research methodology chosen and current status respectively. Section VI – Emerging Observations and Discussion describes the findings to-date followed by Section VII – Conclusion and Future Work, offering thoughts about contribution thus far and what work remains to be done.

II. BACKGROUND AND PROBLEM

By the late 1990s, three forces had taken hold which dramatically changed the nature and challenge of software development. One was the emergence of the Internet which introduced new uses of information technology as well as business models. This, combined with much lower hardware costs, computing capability rapidly appeared on almost every desk and in almost every home. Third, the introduction of graphical user interfaces dramatically enriched user interaction with technology and also complicated software design and development. These three forces together resulted in more software being developed as *products* for a market instead of predominantly bespoke system development that was the norm prior. This shift towards product development introduced substantially more uncertainty into much of the software development activities.

In response to this, a Kuhnian “model revolution” [3] emerged that took a new view on change, risk, and uncertainty in software development. This ‘agile’ approach accepted that requirements could change or that further understanding would emerge throughout the development effort in contrast to more disciplined Software Development LifeCycles that strived to lock down requirements in the specification and planning stages.

The agile model placed greater focus on the development team, recognizing that prescriptive processes were insufficient to ensure project success in these complex and emergent conditions and that the dynamics of the development team, which was now usually cross-functional and empowered to truly own the software product, was considered a critical success factor in delivering software.

While the agile approaches improve many of the issues that were breaking down during the crisis period, many still cling to the notion that there is a customer (or, an internal surrogate), an authoritative voice that the development team can iteratively interact with to clarify requirements and validate results. However, as software solutions address more complex and subtle needs and as development is often more product-oriented, intended for a market rather than a single customer, a new and critical challenge emerges for software teams and that is how to gain a deep understanding of the world for which the product is intended, an understanding that cannot be passed on to the team by an internal market surrogate. Certainly, techniques to ‘hear’ from the market are helpful but, as Polyani [4] noted, market participants have tacit knowledge -- people can know more than they can tell and they know more than can be easily observed.

In early times when requirements were less complex, could be more precisely expressed, and quite often coming from an identifiable customer, techniques such as having at least one domain expert on (or available to) the team were often sufficient. Today, however, with much more technical and problem complexity, heterogenous customer targets, and competitive uncertainties, it is insufficient to simply have one person with this deep understanding, typically creating the requirements specification. Yet, many software development organisations operate this way, often resulting in *requirements fixation* [5].

Rather it is important that everyone on the team have a deep domain understanding. It is also critical that the entire team understands it in a compatible and consistent way because team members (individually, in sub-teams, and across all functional roles) make decisions almost continually based on their individual understanding of the context of the requirements, and much of that context understanding is tacit. This challenged is expressed well by Berry [6] when discussing assumptions in requirements engineering amongst team experts:

“It seems that among experts, a common disease is the presence of unstated assumptions. Because they are unstated, no one seems to notice them. Worse than that, it seems that no two people have the same set of assumptions, often differing by subtle nuances that are even more tacit than the tacit assumptions. It is these assumptions that confound attempts to arrive at consensus, particularly because none of the players is even consciously aware of his or her own assumptions and certainly not of the differences between the players’ assumptions” (p.180)

Thus, product development teams have to strive for a deep *collective* understanding of the context of their product, a shared mental model of the supra-domain, since many decisions are unconsciously made within the team’s understanding of the domain context. Some teams achieve success in this aspect more than others and software development leaders have no theories that help explain why.

We observed earlier [1] that the organisational model surrounding the cross-functional teams has an impact on the team’s ability to grok, hence the scope of this inquiry expands from there to examine additional factors both internal and external to the teams.

III. Research Motivation and Focus

This study aims to develop theory offering insights into factors that support or impede CFPTs in collectively achieving a deep understanding of the context of their products.

The differences between teams that achieve a reasonable degree of collective grokking in terms of team vision, cohesion, and quality of work product is observable by practitioners and researchers, yet the reasons are generally unclear. Without explanatory models, industry leaders are unable to proactively create and nurture the relevant factors. This study is aimed at helping industry practitioners explain why certain prevailing techniques and empirical approaches for understanding software solution needs are often inadequate, why some succeed while others do not.

The focus of this research is practicing software product teams in action, including teams empowered to own their product and those that are not. For contrast, we also include organisations that are not product companies. The study examines the empirical adaptations these teams make toward furthering their understanding of the context in which their users operate. We also examine important organisational factors that either allow or inhibit a team’s ability to collectively grok the domain.

IV. Related Work

We reviewed published material in 3 areas - requirements engineering, design science, and collective sensemaking.

This inquiry is primarily related to requirements engineering (attempting to obtain and understand the true needs). Reviewing all the papers at the IEEE International Requirements Engineering Conference over the past decade, plus many other published papers in the area, we found growing sentiments expressed about the shortcomings of prevailing approaches to requirements engineering which tend to focus on techniques and methods rather than deepening practitioner and team understanding, e.g., (Schon et al. [7], Ralph and Mohanani [8]). This general sentiment led to the formation of the NaPiRE initiative (Naming the Pain in Requirements Engineering) [9], a community endeavour run by a multitude of researchers world-wide. While there are certain domains where the 'techniques and methods' approach is entirely adequate and appropriate, our focus is on problem domains that do not lend themselves well to complete and unambiguous specifications and, therefore, where it is necessary for the CFPTs to have their own deep understanding of the product domain beyond just the requirements specifications.

The design science space has considerable material regarding empathy-driven design (translating human needs to experiences), e.g., (Koppen and Meinel [10], van Rijn et al. [11], Postma et al. [12], Woodcock et al. [13], Dong et al. [14], Kourprie and Visser [15], Kolko [16]). However, we found this falls short of addressing our inquiry question in three critical respects: 1) focus on the design activity as part of an essentially sequential product development process rather than design as part of an on-going continuous product development effort, 2) it tends to focus on the design individual or only the design team rather than the whole development team and, 3) when even the design team is considered, it tends not to be viewed as a *unit* regarding its empathic ability. Design science models described by Wieringa [17] acknowledge the challenge that empathy-driven requirements understanding attempts to address but stops short of suggesting how those challenges could be addressed. We aim to offer insights into how this level of understanding is achieved and how to nurture the pre-requisite conditions.

Collective sensemaking (the process by which people give meaning to their collective experiences) does consider the collective (team) but only with respect to its relationship to the organisation, not to its understanding of an external domain. Of interest in this area is the Cynefin framework (Kurtz and Snowden, [18]) which is a sensemaking framework that is designed to allow shared understandings to emerge which could be insightful with respect to how teams ingest, socialise, and collectively store insights. As with other collective sensemaking models, however, it has resonance in early problem-solving stages and for formal and finite periods of time whereas our focus is on the full product lifecycle.

V. METHOD AND STATUS OF THE RESEARCH

We take an interpretive epistemological stance, employing the Constructivist Grounded Theory qualitative research methodology described by Charmaz [19]. Constructivist Grounded Theory is highly applicable in research such as this because the method is explicitly emergent, taking an inductive approach where no adequate prior theory exists. This method is particularly appropriate for a "What is going on here?" type of qualitative inquiry as this study is. The use of Grounded Theory in computer science research has risen significantly since 2005 and specifically used successfully to study Agile software development teams, e.g. Adolph et al., [20], Dagenais et al., [21], Coleman and O'Connor, [22], Martin, [23], Hoda, [24], Stol et al, [25].

Using theoretical sampling where the analysis of the data collected prior informs the selection of and inquiry with the next participants, individual participants and corporate sites selected are ones involved with software product development (teams developing software for market) and that claim to have cross-functional product development teams. The primary data collection methods are observations of team meetings and team interactions, enriched by semi-structured interviews (recorded and transcribed) with open-ended questions that can allow real issues to emerge. Thus, the method is grounded in the participants' world and the emerging and evolving theory is constructed by the researcher and the participants.

We employ various strategies (Maxwell, [26]) to mitigate threats to validity (credibility, dependability, reliability). Intensive, on-going involvement, e.g., extended participation and the ability to live in the participants' workplace, provides richer data and data types, less dependence on inference, and opportunity for repeated observations and interviews, all which will help rule out spurious associations and premature theories. Participant checks (obtaining participant and peer feedback on the data collected and conclusions drawn) help rule out possibilities of misinterpretation. Select codes and concepts from the analysis are highlighted below as ***bold italics***.

To date, we are working with six software firms. Four of these firms produce commercial enterprise-class software products, one creates sophisticated virtualisation solutions, while another develops large-scale aerospace systems as bespoke system development. Three of these firms have adopted agile as a paradigm, two as a methodological approach, while the other employs a highly prescriptive methodology due to the dictates of its market. The firms range in size from ten to several hundred employees and the firms range in age from 2 to 50 years old. To-date, 18 product development teams across these companies have participated, resulting in 26 individual semi-structured interviews and 19 team observation sessions. The individuals interviewed have been 2 senior managers, 8 senior engineer / team leads, 5 product managers, 1 quality assurance specialist, and 10 intermediate-level software engineers. Participant sampling and data gathering is on-going.

VI. EMERGING OBSERVATIONS AND DISCUSSION

We have identified three contexts that contribute to a CFPT's ability to collectively grok the product requirements. The first is the organisational context which we identified in isolation in our earlier work [1]. The second is the product planning context (the ability of CFPTs to own a broader responsibility for product planning), influenced by the organisational context but having its own independent dynamics. The final context is the product domain context itself (the ability for the CFPT to grok the product domain), heavily dependent on the previous two contexts.

A. The Organisational Context and Its Impact on CFPTs

Fuller [1] described the impact the broader organisational model has on the CFPT team. Impacts of note were intra-team deference, the team's concern and ownership horizon, and the team robustness. Certain aspects of individual participation on the team (e.g. primary affiliation, individual agenda) were also highlighted. In this sub-section, we summarise those findings.

When a functional organisational structure exists in the software product enterprise, e.g., separate engineering, design, product management departments, each contributing individuals to form CFPTs, team members are more likely to limit their contributions to topics directly relating to their area of functional expertise and tend to show marked deference to team members of other functions on topics outside their area of primary functional expertise. The individual sense of primary affiliation was stronger toward their functional department than it was with the software product team. Simply put, an individual in this organisational environment is located via function more than via team membership. This results in team members being much more concerned about *how* a product is to be built and defer to others regarding the *what* and *why*. Illustrative comments from team members were "***I just do what I'm asked to do***" when referring to involvement with requirements specifications or "***They're the experts, I trust them***" when referring to team members in other functional roles.

Team members in this model tended to show less investment in the overall success or failure of the product and the teams themselves much less likely to take collective responsibility for success or failure of the product. They are more likely to shift responsibility to management decisions or to other teams/functions rather than attempt to reconcile differing mandates of the participating functional departments.

In contrast, organisations without a functional structure surrounding the CFPTs seem more likely to have teams with richer intra-team interactions with softer (sometimes an absence of) functional interfaces amongst individual team members, placing the interests of the product foremost and above any functional tensions. In short, the sense of team and commitment to the product tended to be much stronger. In one of our participant companies with multiple products, it is common for product team members' LinkedIn profiles showing the product name as the company they work for with no reference to the overall firm, making it very clear where they belong and what they are committed to.

Studies by Gladstein [27] and Anacona [28] noted that contextual factors have a greater influence on team effectiveness than do internal team processes. Our emerging results to-date support this and suggest further that the two are not unrelated – that the operating context of the team has a significant impact on internal team factors, which include internal team processes.

In summary, a CFPT's progression along a spectrum from an assembly of experts to a true empowered cohesive team is heavily influenced by whether a broader functional department organisational structure exists around the team and how strong those departmental distinctions are.

B. The Product Planning Context and CFPTs

We observed that CFPTs that have strong internal connections and softer functional role deference showed more interest in the broader product planning context. These teams ask broader questions, are more curious, and attempt to explore more - essential ingredients for innovation.

However, our observations also included teams in some companies that did not have the organisational structure and/or culture that allowed their CFPTs to own as much of the product planning process that the teams often wished they could own. This was often the case where strategic planning for product occurred in another functional area and communicated to the product development group to execute upon. Some companies will take this even further and have a separate product management function that define product evolution details that are then handed off to software engineering for development. We observed that CFPTs with strong internal cohesion have a propensity to own *something* and will, therefore, ***narrow*** **or** ***broaden their lens*** on the product development work to match what they are permitted to own. This action of ***Broadening the Lens*** allows the team to identify control boundaries and also to see patterns and relationships so that they may more purposefully and knowledgeably re-focus.

This lens adjustment also aligns their definition of success with what the company expects. Individuals and teams will ***colour within the lines*** they are given or allowed. This is reflected in what completed work the development teams celebrate, e.g. a successful iteration, meeting a release deadline, or being part of a successful product in the market.

The spectrum of this context ranges from full strategic and execution ownership of the product on one end to the team being ***spoon-fed*** tasks on the other.

As with the Organisational Context, the focus of a team's product planning lens also shows in the verbal language used by the teams. The broader the team's planning scope is, the more the conversations will indicate a deep understanding of (or, at least references to) product needs from the domain perspective, product/market opportunities, etc. Teams low on this spectrum reference those considerations less and make more reference to internal entities and artefacts such as other functions/teams, processes, specifications, etc.

C. The Product Domain Context and CFPTs

As Fuller [1] observed, empowered and cohesive CFPTs play a longer game. With less internal deference in the team and less individual tentativeness with respect to their membership on the team, conditions exists that encourage full participation and commitment (both individual and collective) to the long-term product roadmap.

In our analysis to-date, we find that CFPTs that exhibit little to no functional deference across functions within the team and who are not being ***spoon-fed*** their development tasks almost always exhibit *some* degree of collective grokking of the context of their product domain and, hence, the product requirements.

This is significant because all software is developed in context and it is context that guides decisions. If the team is cohesive, their context will be more collective than if it is not (Organisational Context). If the team owns more of the product planning, that context will be more comprehensive than if they own less (Product Planning Context). And if they collectively grok the product domain to a reasonable degree, their context will more accurately reflect the world for which their product is intended (Product Domain Context).

The spectrum of collective domain understanding ranges from ***just do what the story says*** to intellectual domain understanding (deep knowledge of vocabulary, workflows, objectives, etc.) to true felt (lived) understanding of the domain. The further a team moves along this spectrum, the more the team groks - blurs the boundaries between itself and the domain in order to achieve some degree of empathic understanding.

In this context of requirements engineering, we suggest that empathy, specifically collective cognitive empathy, is a fundamentally important ability in order to deeply understand a domain which the team is otherwise unfamiliar with. Exercising that ability, stepping into that other domain, involves a certain temporary softening of the distinction between the collective and the domain, a ***blurring of the boundaries***, in order to truly understand perspectives in that domain. ***Broadening the lens*** is necessary for the team to be able to see the other domain and its context, the ***blurring of the boundaries*** is an effort to understand. Smith et al. [29] suggest that empathy can become collective and that it can be an attribute of the group that is more than just the aggregation of individuals' attributes.

We observed some teams that did not even attempt to grok the product domain, a reflection of the culture of the team and its organisational environment. Certain other teams that did try had modest success due to influences from the organisational and/or product planning contexts.

For a CFPT to be able to *collectively* step into another domain, it is necessary for it to see itself as a cohesive unit. This can only be achieved when there is a high level of transparency across all functions on the team, little to no deference shown within the team, and a strong sense of collective ownership for the product. In other words, a true *team* with a strong product mandate – blurred boundaries with strong connections. It requires team members to feel psychologically safe, have open minds, and a strong sense of curiosity. If any of these are weak or missing, discoveries, innovation, and collective grokking are inhibited [30].

As we examined the teams that made some progress at collective grokking of the product domain, we observed a special form of the ***broadening the lens*** behaviour that teams performed when refining their product planning context. In these cases, the teams were *purposefully* ***blurring boundaries*** in order to achieve a deeper collective empathic understanding of the product domain.

VII. Conclusion and Future Work

Many of these observations sit in opposition to common organisational practices that emphasise specialisation (for management and control convenience) and focus (to meet deadlines). Further work is needed to bring more clarity about whether there are other, more subtle, factors at play.

Our data strongly indicates that ***blurred boundaries*** within CFPTs are a reflection of blurred boundaries outside of the teams and, similarly, there may well be even further team environmental factors to explore.

There appears to be a certain blurring of functional and domain boundaries necessary for a team to become a true product team rather than a collection of functional experts assembled around a product. Further, this appears to be a pre-condition for the team to be able to behave as a collective and achieve some degree of collective grokking of the context of the product requirements.

While we observed teams using the ***broadening the lens*** mechanism to ***blur the boundaries*** between the team and the domain, we allow that this mechanism and the pre-requisite or enabling conditions may paint only a partial picture. Thus, we believe there remains much to explore with respect to why some teams, even in the same organisational context, observably achieve more grokking of the product domain than do other teams.

Acknowledgment

This work is supported in part by the Institute for Computing, Information and Cognitive Systems (ICICS) at UBC.

References

[1] R. Fuller, "What T-shirt Are You Wearing? Towards the Collective Team Grokking of Product Requirements," in *SOFTENG 2019, The Fifth International Conference on Advances and Trends in Software Engineering*, pp. 37–40, 2019.

[2] R. Krznaric, *Empathy: why it matters, and how to get it*. New York: Penguin Random House, 2014.

[3] T. S. Kuhn, *The Structure of Scientific Revolutions*. 4th ed. University of Chicago Press, 2012.

[4] M. Polanyi, *The tacit dimension*. Chicago: University of Chicago Press, 2009.

[5] R. Mohanani, P. Ralph, and B. Shreeve, "Requirements Fixation," in *Proceedings of the 36th International Conference on Software Engineering*, pp. 895–906, 2014.

[6] D. M. Berry, "The importance of ignorance in requirements engineering," *Journal of System Software*, vol. 28, no. 2, pp. 179–184, 1995.

[7] E. M. Schön, D. Winter, M. J. Escalona, and J. Thomaschewski, "Key challenges in agile requirements engineering," in *Lecture Notes in Business Information Processing*, 2017.

[8] P. Ralph and R. Mohanani, "Is Requirements Engineering Inherently Counterproductive?," in *Proceedings - 5th International Workshop on the Twin Peaks of Requirements and Architecture, TwinPeaks 2015*, 2015.

[9] D. Mendez, S. Wagner, M. Kalinowski, M. Felderer et al.. NaPiRE: Naming the Pain in Requirements Engineering, http://napire.org.

[10] E. Koppen and C. Meinel, "Knowing People: The Empathetic Designer," *Design Philosophy Papers*, vol. 10, no. 1, pp. 35-51, 2012.

[11] H. Van Rijn, F. S. Visser, P. J. Stappers, and A. D. Özakar, "Achieving empathy with users: the effects of different sources of information," *CoDesign*, vol. 7, pp. 65–77, 2011.

[12] C. Postma, E. Zwartkruis-Pelgrim, E. Daemen, and J. Du, "Challenges of Doing Empathic Design: Experiences from Industry," *Int. J. Des*. Vol 6, No 1, pp. 59-70, 2012.

[13] A. Woodcock, D. McDonagh, J. Osmond, and W. Scott, "Empathy, Design and Human Factors," *Advances in Usability and User Experience*, pp. 569-579, 2018.

[14] Y. Dong, H. Dong, and S. Yuan, "Empathy in Design: A Historical and Cross-Disciplinary Perspective," *Advances in Neuroergonomics and Cognitive Engineering*, pp. 295-304, 2018.

[15] M. Kouprie and F. S. Visser, "A framework for empathy in design: stepping into and out of the user's life," *J. Eng. Des.*, vol. 20, no. 5, pp. 437–448, 2009.

[16] J. Kolko, *Well-Designed: How to create empathy to create products people love*. Harvard Business Review Press, 2014.

[17] R. Wieringa, *Design Science Methodology for Information Systems and Software Engineering*. Springer, Berlin, 2014.

[18] C. F. Kurtz and D. Snowden, "The New Dynamics of Strategy: Sense-making in a Complex-Complicated World," *IBM Syst. J.*, vol. 42, no. 3, pp. 462–483, 2003.

[19] K. Charmaz, *Constructing grounded theory* (2nd ed.). London: Sage, 2014.

[20] S. Adolph, W. Hall, and P. Kruchten, "Using grounded theory to study the experience of software development," *Empirical Software Engineering*., vol. 16, no. 4, pp. 487–513, 2011.

[21] B. Dagenais, H. Ossher, R. K. E. Bellamy, M. P. Robillard, and J. P. De Vries, "Moving into a New Software Project Landscape," in *ICSE '10 Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering*, pp. 275–284, 2010.

[22] G. Coleman and R. O'Connor, "Using grounded theory to understand software process improvement: A study of Irish software product companies," *Information Software Technology*, vol. 49, no. 6, pp. 654–667, 2007.

[23] A. M. Martin, "The Role of Customers in Extreme Programming Projects," PhD thesis. Victoria University of Wellington, New Zealand, 2009.

[24] R. Hoda, "Self-Organizing Agile Teams : A Grounded Theory," PhD thesis. Victoria University of Wellington, New Zealand, 2011.

[25] K. J. Stol, P. Ralph, and B. Fitzgerald, "Grounded theory in software engineering research: A critical review and guidelines," in *Proceedings – International Conference on Software Engineering*, vol 14-22. pp. 120-131, 2016.

[26] J. A. Maxwell, *Qualitative research design: An interactive approach*. Thousand Oaks, Calif.: SAGE Publications, 2012.

[27] D. L. Gladstein, "Groups in Context: A Model of Task Group Effectiveness," *Adm. Sci. Q.*, vol. 29, no. 4, p. 499, Apr. 2006.

[28] D. G. Ancona and D. F. Caldwell, "Demography and Design: Predictors of New Product Team Performance," *Organ. Sci*., vol. 3, no. 3, pp. 321–341, Oct. 2008.

[29] E. R. Smith, C. R. Seger, and D. M. Mackie, "Can Emotions Be Truly Group Level? Evidence Regarding Four Conceptual Criteria,". *J. Pers. Soc. Psychol*., 2007.

[30] M. Harms and R. Reiter-Palmon. *Team Creativity and Innovation*, Oxford: Oxford University Press, 2018.