ABSTRACT
Software engineering needs a general theory, i.e., a theory that applies across the field and unifies existing empirical and theoretical work. General theories are common in other domains, such as physics. While many software engineering theories exist, no general theory of software engineering is evident. Consequently, this report reviews the emerging consensus on a general theory in software engineering from the Second SEMAT General Theory of Software Engineering workshop co-located with the International Conference on Software Engineering in 2013. Participants agreed that a general theory is possible and needed, should explain and predict software engineering phenomena at multiple levels, including social processes and technical artifacts, should synthesize existing theories from software engineering and reference disciplines, should be developed iteratively, should avoid common misconceptions and atheoretical concepts, and should respect the complexity of software engineering phenomena. However, several disputes remain, including concerns regarding ontology, epistemology, level of formality, and how exactly to proceed with formulating a general theory.

Keywords

1. INTRODUCTION
The General Theory of Software Engineering (GTSE) initiative promotes theory development and theory-driven empirical research on all aspects of software engineering. It aims to eventually produce a GTSE, i.e., a theory that broadly explains software development phenomena, unifies existing theory, facilitates a cumulative research tradition and supports Software Engineering (SE) education and practice. This report summarizes breakthroughs from the 2013 GTSE Workshop (GTSE ’13).

In this report, the term “software engineering” is used broadly to refer to all activities involved in conceptualizing, creating and modifying software intensive systems. A theory is simply a collection of interconnected ideas intended to explain, describe, analyze or predict some phenomena. A general theory is a theory that applies to a broad range of phenomena, across several levels of analysis, or consolidates several theoretical perspectives.

One theme that emerged during GTSE ’13 was the necessity to build consensus around the need for, scope of, and composition of a GTSE. Adopting a consensus approach should increase not only the initial GTSE’s quality (by integrating the ideas of many participants) but also its palatability (as participants come to support a theory they view as encapsulating their own ideas). Moreover, simply proposing a limited initial theory may attract criticism for either being too general (“another theory of everything”) or confined to a single perspective. Such initial and limited theories provide a starting point, which can evolve into better and sounder theories.

Consequently, this paper explores the emerging consensus, and limits thereof, concerning properties of a GTSE (Section 2). We then briefly describe the history and structure of the workshop (Section 3) and offer some thoughts on the future of the GTSE project (Section 4).

2. EMERGING CONSENSUS ON A GTSE
The presented papers and ensuing discussion revealed many areas of consensus and several remaining disputes concerning GTSE. These themes both reaffirm and build on the five needs identified in the GTSE 2012 workshop – (1) sound theoretical foundations for SE, (2) diverse theoretical approaches for formulating a GTSE, (3) consensus on a primary dependent variable (possibly Software Engineering Success), (4) better metrics and instruments for SE variables, and (5) better descriptive research [14, 12].

2.1 Agreements
While some academics may react skeptically to the possibility of a general theory of software engineering, general theories are quite common in all branches of science [9]. Well known general theories include the Standard Model (physics), the Periodic Table of Elements (chemistry), Big Bang Theory (cosmology), the Theory of Evolution (biology), Structuration Theory (sociology), Supply and Demand (microeconomics), the General Theory of Employment, Interest and Money (macroeconomics), the General Theory
of Crime (criminology) and the Theory of World Conflict (political science).

Participants agreed that general theories are neither unusual nor suspicious and that SE has no unusual property that should preclude general theory. Participants agreed that theorizing takes many forms [17] and SE entails myriad phenomena; for instance, Perry [11] distinguishes between software engineers, software engineering and software project management. Ralph [12] consequently suggests formulating a multi-level GTSE, i.e., a theory that crosses many units of analysis including individual, team, artifact, process and project. A core question then is: What might the different levels of a GTSE contain?

Building on the previously identified need for a clear dependent variable [14], Ekstedt [3] suggests that a GTSE should identify the primary drivers of Software Engineering Success. Furthermore, a GTSE should also explain the social process by which software is created [12] including software practices [16] and the personal values of participants [1]. Moreover, a GTSE should also encompass automated software design—especially given increasing possibilities for automation into the future [2]. Additionally, clear and agreed terminology is needed to facilitate communication and understanding of the GTSE [4, 10].

More generally, a GTSE may incorporate several existing theories from SE reference disciplines. Erbas and Erbas [4] suggest Transaction Cost Economics as a possible theoretical foundation for explaining why developers adopt different approaches to SE. Similarly, Smolander and Päivärinta [16] recommend Reflection-in-Action as a theoretical framework for the design process. Meanwhile, Ralph [12] suggests several theories at different levels of analysis—Complexity Theory (project), Sensemaking-Coevolution-Implementation Theory (process), Boundary Objects (artifact), Transactive Memory (team) and Cognitive Bias (individual).

There are many viable ways of approaching GTSE development. Adolph and Kruchten [1] adopt a grounded theory approach. Others focus on adapting or extending existing theory [12, 16, 4]. Others take a more rationalistic approach [11, 2]. While the best approach is not clear (as discussed below), it is clear that inherent complexity of formulating a GTSE necessitates iterative theory development [3].

Taking a different perspective, we can also ask what mistakes or misconceptions a GTSE should avoid? Smolander and Päivärinta [16] warn that GTSE should be based on observations of real-world practice to avoid idealized or otherwise inaccurate assumptions. Meanwhile, Exman examines four specific dangers, e.g., GTSE should respect the emergent properties of running software, i.e., properties of running software not evident from static source code. More generally, SE appears replete with isolated and incomplete theories. Participants agreed that general theories are neither unusual nor suspicious and that SE has no unusual property that should preclude general theory.

Participants also brought differing conceptualizations of the present state of the field. Exman [5] for example views the field as being pre-paradigmatic while Stol and Fitzgerald [17] characterize SE as an archipelago of loosely-coupled theory fragments depending on comparatively testing rival theories. In this way, knowledge becomes the best existing theory rather than a justified, true belief. However, this too is problematic for evaluating a GTSE as an appropriate rival theory is not evident.

A possibly related theme concerns the desirable level of formality. Physical science theories are often expressed as mathematical laws, e.g., relativity, Maxwell’s equations. However, social science contexts often resist such formal descriptions due to their multifarious, probabilistic causal webs. Therefore, it is unclear how formal a GTSE should be, or to what extent the desirable level of formality varies across units of analysis. Perhaps, for example, artifact properties may be described more precisely than team dynamics. While some participants (e.g. [2]) suggested more algebraic descriptions, others are concerned that attempts to increase formality may lead to more idealistic, less empirically valid and practically usable theory.

### 3. HISTORY AND STRUCTURE OF THE WORKSHOP

After a successful first workshop in Stockholm, Sweden [14], GTSE 13 was held on May 26th in conjunction with the International Conference on Software Engineering, ICSE 2013, in San Francisco [9]. The GTSE workshops are organized by SEMAT (Software Engineering Methods and Theory), an informal organization founded by Ivar Jacobson, Bertrand Meyer and Richard Soley to make the work and results from industry, research and education more relevant to one another and thereby to the state of software engineering. SEMAT organizes its efforts in two areas—the theory area and the practice area.

The practice area strives to establish a set of widely agreed elements to describe software engineering and its practices. To this end, the practice area has submitted a standard proposal, known as “Essence” [9], to the Object Management Group (OMG).

The theory area, headed by Michael Goedicke and Pontus Johnson, initiated the GTSE work when Mathias Ekstedt and Pontus Johnson joined the SEMAT initiative after writing extensively on the potential for general theory in SE [7]. The theory area core argument is that while the SE field has produced many theories, no general or unifying theory is evident; however, a GTSE is both possible and desirable [8]. The theory area has since organized the 2012 and 2013 GTSE workshops, and a special issue of *Science of Computer Programming* on GTSE is being planned.

The aim of the GTSE initiative and workshops is to promote and facilitate the scientific process of proposing, debating, testing and revising general theories of SE. The implicit goal is to push SE toward a state where one or a few theories constitute the scientific core of the field and provide communicable knowledge and accurate predictions of central SE phenomena. Consequently, the workshop called for papers proposing aspects of a GTSE or discussing questions including:

- How can a general theory of SE be of practical use?
- What are the objectives of such a theory?
- What questions should it address?
- What is a useful definition of theory?
- How foundational/universal should a general theory of SE be?
- What should its main concepts be?
- How formally or informally should it be expressed?

The workshop received 26 submissions, each of which underwent at least three and on average four reviews. Based on the reviews, ten of the papers were accepted. The accepted papers considered diverse aspects of software engineering theories including mathematical, engineering, management and sociological.

The workshop proceeded in three parts:

1. Introduction by the organizers and review of general theories in other disciplines;
2. Paper presentations;
3. Open discussion and consensus building.

For consensus building, participants first voted on which question to discuss and then discussed questions in descending order of popularity until time ran out. The questions discussed were as follows:

- Do we agree on the purpose of a general theory of SE? What would it be good for?
- Should we look for underlying theories of SE? What could they be? Social theories, technical theories, economic theories?
- Which approach is the most practical?
- Should it be expressed formally? If formalized, what is a suitable language?
- How to evaluate theories, meta-theories?
- Evolution
- Challenges in generating GTSE?
- What questions and subquestions should it address? What should its main concepts be?
- How to build a GTSE? Starting from domain-theory?
- Is general theory meta-theory?
- How foundational/universal should a general theory of software engineering be?
- Sampling
- What is a useful definition of theory?

---

4. CONCLUSIONS

In summary, the second SEMAT General Theory of Software Engineering workshop was very successful. The relatively large number of submissions suggests there is considerable interest from the SE research community in this topic. The papers and open discussion clearly built on the previous workshop and furthered the GTSE formulation process. It became clear at this workshop that a consensus approach was needed and participants reached several areas of consensus, including the following:

- General theories are common across physical and social sciences.
- A GTSE should explain myriad SE phenomena across several levels of analysis (individual, team, artifact, etc.).
- A GTSE should address both the process of SE and the antecedents of key variables including software engineering success.
- A GTSE may incorporate theories and theory fragments from SE and reference disciplines.
- A GTSE should respect the complexity of SE phenomena.

In addition to formulating a GTSE, future research may address several remaining disputes, including the appropriate epistemology for a GTSE and the desirable level of formality. More research on the dimensions and measurement of software engineering success is also needed. Finally, we invite submissions to and participation in future GTSE workshops. Following this year’s success, we plan to organize GTSE 2014 as an ICSE workshop, i.e., in Hyderabad, India in June.

5. ACKNOWLEDGEMENTS

We extend our thanks to the participants of the workshop:

Alexey Nikitine, Arbi Ghazarian, Brian Fitzgerald
Cengiz Erbas, Dewayne Perry, Don Batory
Hausi Müller, Iaakov Exman, Ira Baxter
Ivar Jacobson, Kari Smolander, Karl Reed
Klaas-Jan Stol, Konstantin Weitz, Magno Cavalcante
Marco Kuhrmann, Mathias Ekstedt, Michael Goedicke
Miguel Trujillo, Mira Kajko-Mattsson, Pan-Wei Ng
Paul Ralph, Philippe Kruchten, Pontus Johnson
Shihong Huang, Steve Adolph, Tero Päivärinta

6. REFERENCES


