Understanding Interaction and Communication Challenges Present in Software Engineering

Sergey Masyagin, Giancarlo Succi[a], Sofiia Yermolaieva[b] and Nadezhda Zagvozkina
Innopolis University, Russia

Keywords: Communication in Software Engineering, Verbal and Nonverbal Communication, Systemic Theory, Democratic Theory.

Abstract: Researchers have largely identified that interactions and communications pose major challenges in software development, especially when extracting requirements. However, they have not appreciated the sources and the depth of them, thus approaching them with mechanisms that have not (fully) achieved the desired objectives. In this position, we claim that such challenges can be explained using three major theories coming from social sciences: the theory of verbal and nonverbal communication, systemic theory, and democratic theory. We also argue that some of the successful practices of agile methods can be explained in terms of these theories. Finally, we stipulate that a full appreciation of these theories can result in a significant leap forward in the discipline, identifying new mechanisms that can help to overcome the mentioned challenges, understanding fully what we are doing and why.

1 INTRODUCTION

Software engineering has been plagued since its early days by what has been called the "Software Crisis" that is, the rapid increase of computational power of hardware machines and complexity of approachable problems in parallel of no adequate methods to approach the creation of the solution to such problems (Science Committee, 1968). Despite more than 50 years of work, the problem still exists and has been renewed. In the Chaos report conducted in 2015, it was identified that only 29% of projects are successful by the above-mentioned criteria (Standish Group, 2015). Fitzerald (Fitzgerald, 2012) redefined the current problems as "Software Crisis 2.0". This crisis is based on the rise in both advanced hardware technologies and a huge amount of available data, which opens various opportunities for customers while at the same time cannot be addressed by the software due to its limited capabilities. Both crises could be explained with the Wirth's law that covers differences in the evolution of software, that is getting slower much faster comparing to the increasing evolution's speed of hardware (Wirth, 1995).

A core aspect of the software crisis is linked to user involvement, incomplete and changing requirements which have been largely considered one of the core factors that cause projects to fail both in the various version of the Standish Report (Standish Group, 2014) and in several other quite authoritative sources, also in grey literature (360logica, 2019; Mooney, 2018). Rice and Perry state that unclear requirements lead to the development of low-quality software (Rice and Perry, 2011).

These kind of problems were among the key motivation for the development of agile methodologies a couple of decades ago (Beck et al., 1999; Cockburn and Highsmith, 2001). All the different agile approaches promote the interaction inside teams and among teams, customers, and (other) key stakeholders.

However, all of the above-mentioned challenges still exist. Moreover, indeed agile methods have undoubtedly contributed to the improvement of the development process. However, even if they are now overwhelmed accepted, root causes of such problems have not been studied, but just proposed solutions that have been verified to empirically work in several case (Ceschi et al., 2005; di Bella et al., 2013).

In this proposal, we want to present a framework to describe such problem and also, to be provocative, argue why we think that some of the well-established approaches to address the software crises simply can-

[a] https://orcid.org/0000-0001-8847-0186
[b] https://orcid.org/0000-0002-3198-6330

Masyagin, S., Succi, G., Yermolaieva, S. and Zagvozkina, N.
Understanding Interaction and Communication Challenges Present in Software Engineering.
DOI: 10.5220/0009581905720578
In *Proceedings of the 15th International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE 2020)*, pages 572-578
ISBN: 978-989-758-421-3

not work.

We root our position in three theories coming from social sciences: the theory of verbal and nonverbal communication, systemic theory, and democratic theory.

We claim that such theories can explain the challenges faced in interactions and communications while developing software. Moreover, we evidence that the progress made by agile methods are clear consequences of these theories. Finally, we stipulate that a full appreciation of these theories can result in a significant leap forward in the discipline, identifying new mechanisms that can help to overcome the mentioned challenges, understanding fully what we are doing and why.

This paper is organized as follows. Section 2 present the background theories of our investigation, and, in particular, Section 2.1 presents the differences between verbal and nonverbal communication, Section 2.2 outlines systemic theory, and Section 2.3 summarised the principles of democratic theory. Sections 3, 4, and 5 analyze the implication of verbal and nonverbal communications, systemic theory, and democratic theory, respectively to software development. Section 6 summarizes the results of our position, and Section 7 draws some conclusion and outlines our future work in this area.

2 BACKGROUND

This section outlines the theories that we have used to develop our position. Specifically, we are going to cover:

- verbal and nonverbal communication,
- systemic theory,
- democratic theory.

2.1 Verbal and Nonverbal Communication

The study of how people communicate started with the work of Darwin (Darwin, 1873) about 150 years ago and has taken a major vigor first with the research of Chapple (Chapple, 1939) and more recently with the investigations of Kendon (Kendon, 1967).

It is now clear that interaction streams occur typically along two major channels: verbal and nonverbal (Archer and Akert, 1977; Bugental et al., 1970), and that there are concepts that cannot be explained by verbal communication alone (DiMatteo et al., 1980). Independently from the channel, the interaction consists of two major sub-processes: message production and message reception (Buck and VanLear, 2002). Both of the sub-processes are peculiar for communication, which is a part of interactions, and three major types of communication can be identified:

1. spontaneous nonverbal communication via gestures and body language, which is involuntary and biologically-based,
2. symbolic communication, that is intended, is based on common socially defined behavior and held via a verbal channel,
3. pseudo-spontaneous communication, that is done intentionally by the sender but used as a manipulation to show that it is spontaneous for the receiver.

Nonverbal communication uses feelings and emotions rather than verbal language. Hans and Hans (Hans and Hans, 2015), in their paper, identified that this channel consists of kinesics, head movements and posture, and haptics.

Kinesics covers all human movements, including all types of gestures, such as:

- emblems (gestures with agreed-on meaning),
- adaptors (biological or unconscious touches directed toward self or other object),
- illustrators (representational accompaniment of a verbal message) (Hans and Hans, 2015).

Head movement and posture, according to Hans and Hans, except of obvious parts covered in the name of this subfield of nonverbal communication, also contain eye contact and facial expression.

Haptics focuses on conscious touches with emotional context, along with their intensity and frequency.

There also exists another arguable component of this channel - proxemics. While Hans and Hans identified proxemics as a unit of haptics, other authors such as Hall et al. (Hall et al., 1968) consider it as a distinct type of nonverbal communication. Proxemics focuses on social space and distance as a factor that influences communication.

When diving into the depth of verbal communication, we can see that it has a much simpler nature comparing to nonverbal's one. According to Salzmann (Salzmann, 1998) study of language and culture, we can conclude that this channel could be divided into:

- vocal - spoken language,
- and nonvocal - verbal communication conducted using sign languages, written communication, etc.

What combines these two channels of communication is - semiotics, that is a study of relationships between all types of verbal and nonverbal communication (Cobley, 2001).

2.2 Systemic Theory

Systemic theory aims at representing groups or societies of individuals or of other entities as a set of interrelations between components, where the emerging behavior is more than a planned sum of the behaviors of the individuals, so that we cannot easily decompose the interested reality into parts.

The theory was originally conceived by von Bertalanffy (Von Bertalanffy, 1972), and then evolved in several directions, including biology, psychology, sociology, engineering, etc.

It is particularly interesting the work that Bateson has made in this area (Bateson, 1972). He states that a common knowledge and so understanding does not exist, because each knowledge requires interpretation (Patton and McMahon, 2014). Every individual uses their own interpretation framework, which is based upon background experience and so hypothesis built upon the experience. The opposite to the interpretation process is representation. To transfer any knowledge or information using language, art, or etc. people tend to use the same framework and settled patterns the understanding. Representing and understanding are typical community processes.

Another significant aspect of the systemic theory that should be considered is a near decomposability of entities. Simon (Simon, 1969) states that this concept is used to develop hierarchical systems with independent subsystems that will have only aggregate dependencies on the other subsystems. In the society with high variability of entities (members), near decomposability can help to develop an organization, that won't be affected in case of problems with any entity or its change to another one (Wagner and Altenberg, 1996). As an opposite structure, we can conclude that in non-decomposable systems, all entities have a significant impact on each other and the overall system. Any problem with an entity in such a system will lead to critical issues.

2.3 Democratic Theory

Democratic Theory is a field of political theory that aims to consider society, choices they make, and face along with their consequences (Ansolabehere, 2001). There are various approaches to democratic theory, Dahl (Dahl, 1956) identified the following:

- Madisonian,
- Populistic,
- Polyarchal,
- Equality, Diversity and Intensity,
- and American Hybrid.

Besides different approaches to democratic theory there are different democracy types to analyze. In the paper, we consider participatory democracy which bases on social dialog and collaboration (Moote et al., 1997).

One of the characteristics of a participatory and democratic community is an involvement of all members into a rational public opinion development. All society members participate and contribute to decision making process (Moote et al., 1997). One of the methods to achieve such meaningful involvement is equivalent enlightenment. According to Lasswell (Lasswell, 1948), equivalent enlightenment is the same level of attention and understanding to the specified goal between layman, experts, and leaders. He also states that equivalent commitment in the democratic society is achieved by setting down a common goal and its equal wide-spreading by equivalent enlightenment among all social groups.

3 VERBAL AND NONVERBAL COMMUNICATION IN SOFTWARE ENGINEERING

We have often heard that a key problem in requirement is that a human does not express their requirements in a non-ambiguous way. Therefore we need to define ways to be more stringent in such definition, so to limit such problems.

As an example of this it is enough to check the works of highly cited software engineering scholars, like, for instance, Berry, from (Berry and Berry, 1983) to (Hadar et al., 2019), Finkelstein from (Finkelstein, 1991) to (Finkelstein, 2013), Mylopolous from (Greenspan and Borgida, 1982) to (Guarino et al., 2019), again just mentioning in alphabetic order three outstanding researchers in software engineering throughout their career. Moreover, in a recent work, it has been evidenced that the vast majority of the research in the field is concentrated in trying to represent the requirements formally (Ivanov et al., 2017).

This approach has already been partially contrasted with the approaches present in agile methods. There, on one side, it is clearly said that changes in requirements are welcomed – the first book of Kent Beck on eXtreme Programming contains as subtitle "Embrace Change" (Beck, 1999). On the other side, it is claimed that more precise requirements are achieved via more direct interactions between developers, customers, and managers – in the Agile Manifesto, it is written "Individuals and interactions over processes and tools" (Beck et al., 1999).

However, we think that we can go further. We claim that addressing requirements ambiguity via always more formal methods ignore the fact that we, humans, communicate not only verbally but also nonverbally. And as it has been explained previously in Section 2.1., not all the nonverbal communication can be expressed in terms of verbal communication. As such, the formal approach is intrinsically flawed. On the contrary, trying to involve in face-to-face meetings, customers, managers, and developers does address the problem since it extends to modes of communication, and such an ampler communication spectrum may help substantially in reducing ambiguity.

Needless to say that, anyway, since requirements are elicited and understood in social structures, they are intrinsically volatile. More on this in Section 4

4 SYSTEMIC THEORY IN SOFTWARE ENGINEERING

In addition to the problem of verbal and nonverbal communication, a further explanation of the ambiguous interpretations of requirements can come from how they are actually elicited, taking advantage of systemic theory.

For instance, it is not uncommon that the needs of the same customer are extracted by different analysts. Systemic theory evidences that understanding is a social process that relates to how people interact (Bateson, 1972). Indeed, under such perspective, requirements extracted individually or in pairs (customer,analyst) are intrinsic flawed.

If we compare our approach to art, then when making their masterpieces artists typically use their own way of representation and the spectators of such works dive into the mind of the artists to understand it – this is often not an easy process and requires "interpreters," and actually there could be lengthy discussions what should be the authentic interpretation. Therefore, a person who reads the requirement uses his/her way to understand (interpret) it.

The process could be compared to the encoding and decoding of the information by two systems using different algorithms. Altogether, ambiguity in requirements may also happen because of what systemic theorists call "cognitively inconsistent behavior and thoughts" of the people involved in the process of their development and interpretation since they belong to disjoint communities of meaning(Bateson, 1972).

Systemic theory is also very important to understand the composition of development teams. Let's consider a software organization as a system with entities and relations among them, which follows a software development life-cycle (Ruparelia, 2010), which may differ locally in the different components (a.k.a. subsystems) of the organization but share the same paramount goal (Pressman and Maxim, 2014). Such a goal is commonly documented as requirements and specification documents, which further will be used for development and verification.

According to the systemic theory, such a system could be decomposable if its different subsystems can handle every phase of the software life-cycle. However, if some subsystems are unable to do so, the system becomes not decomposable or only partly decomposable. Indeed, most systems made of knowledge workers tend to be at most only partially decomposable. The only partial decomposability has a huge implication in the division of work, which is, indeed, based on communication and interactions between key stakeholders. Such implications also extend to the individual artifacts of the process developed at every phase (Gasevic et al., 2009).

Indeed, the adoption of agile methods implies a higher level of decomposability, since there is a higher level of sharing and of communication, as we have previously discussed.

5 DEMOCRATIC THEORY IN SOFTWARE ENGINEERING

In a software organization as a democratic society, people are involved in a decision-making process. In organizations that follow agile-based approaches to software development team itself performs decision making, discussion, and accomplishing itself (Basahel, 2014) (Wan and R., 2010).

An interesting question is why we should involve people in the decision-making process? To answer this question, we should dive into the process itself. In a discussion, all members suggest their opinions, which are further argued, and as a result, they make the optimal or the best-suited decision. The critical point of the whole decision-making is not only an identification of a solution to some problem but an involvement of every member in its development, so that shared responsibility will be achieved.

Besides, collective responsibility, that guarantee commitment to the project, decision making can also be considered a technique for equivalent enlightenment as all members will be in the same context and will have a compatible level of attention.

So, if to go back to the problem of "unclear requirements" we can conclude that the same level of immersion into the context through the development

Table 1: Theory and problems it explains.

Theory	Problem
Verbal and nonverbal communication	• Written document (verbal) cannot fully express the meaning of requirements, since there is a need of a nonverbal communication channels.
Systemic Theory	• Software organization are only quasi decomposable, therefore, requirements are intrinsically volatile. • "Cognitively inconsistent behaviour and thoughts" of all the stakeholders and developers involved in the definition of requirements may cause them not to be properly understood.
Democratic Theory	• Different level of enlightenment among people who operate with requirements can make such requirements ambiguous.

of equivalent enlightenment among people who operate in any way with requirements could solve it.

6 SUMMARY OF THE POSITION

A software crisis exists, and we can not argue with a set of existing challenges that slower evolution of the whole software engineering sphere, but we can try to go deep into every particular issue to understand its roots. An understanding is the first and the hardest step towards the development of a solution, that is why we are trying to look at the challenges described in Section 1 from the various angles. We considered the issues from the perspective of the social sciences. The results of our analysis on problems and theories that explain them are presented in Table 1.

7 CONCLUSIONS AND FURTHER WORK

In this paper, we have outlined three theories of social sciences that can explain the problem in interactions and communication present in software development, and we have seen how such theories can explain some of the advantages of agile methods. To move this work from a position to a full research we need now to validate systematically (some of) our findings. We plan, therefore, to launch a session of focus groups and of interviews with software developers, managers, and customers on this topic. To this end, we plan to take advantage of the strategic location of Innopolis University, which is surrounded by a large and rich variety of software companies located inside the city of Innopolis.

ACKNOWLEDGMENTS

We thank Innopolis University for generously funding this research.

REFERENCES

360logica (2019). What are 5 common problems in the software development process?

Ansolabehere, S. (2001). Political Advertising. *International Encyclopedia of the Social & Behavioral Sciences* , pages 11624–11628.

Archer, D. and Akert, R. (1977). Words and everything else: Verbal and nonverbal cues in social interpretation. *Journal of Personality and Social Psychology*, 35:443–449.

Basahel, A. (2014). Adopting scrum agile project management for managing academic institutions. *BIJIT - BVICAM's International Journal of Information Technology*, 7:795–798.

Bateson, G. (1972). *Steps to an Ecology of Mind: Collected Essays in Anthropology, Psychiatry, Evolution, and Epistemology*. University of Chicago Press.

Beck, K. (1999). *Extreme Programming Explained: Embrace Change*. Addison-Wesley Longman Publishing Co., Inc., USA.

Beck, K., Beedle, M., van Bennekum, A., Cockburn, A., Cunningham, W., Fowler, M., Grenning, J., Highsmith, J., Hunt, A., Jeffries, R., Kern, J., Marick, B., Martin, R. C., Mellor, S., Schwaber, K., Sutherland, J., and Thomas, D. (1999). Manifesto for Agile Software Development.

Berry, D. M. and Berry, O. (1983). The programmer-client interaction in arriving at program specifications: guidelines and linguistic requirements. In *Proceedings of IFIP TC2 Working Conference on System Description Methodologies*.

Buck, R. and VanLear, C. A. (2002). Verbal and nonverbal communication: Distinguishing symbolic, spontaneous, and pseudo-spontaneous nonverbal behavior. *Journal of Communication*, 52:522–541.

Bugental, D. E., Kaswan, J. W., and Love, L. R. (1970). Perception of contradictory meanings conveyed by verbal and nonverbal channels. *Journal of Personality and Social Psychology*, 16:647–655.

Ceschi, M., Sillitti, A., Succi, G., and De Panfilis, S. (2005). Project management in plan-based and agile companies. *IEEE software*, 22(3):21–27.

Chapple, E. D. (1939). Quantitative analysis of the interaction of individuals. *Proceedings of the National Academy of Sciences of the United States of America*, 25(2):58.

Cobley, P. (2001). *The Routledge companion to semiotics and linguists. (1st ed.)* . London: Routledge.

Cockburn, A. and Highsmith, J. (2001). Agile software development, the people factor. *Computer*, 34(11):131–133.

Dahl, R. A. (1956). *A Preface to Democratic Theory* . University of Chicago Press.

Darwin, C. (1873). *The expression of the emotions in man and animals* . London: John Murray publisher.

di Bella, E., Fronza, I., Phaphoom, N., Sillitti, A., Succi, G., and Vlasenko, J. (2013). Pair programming and software defects–a large, industrial case study. *IEEE Transactions on Software Engineering*, 39(7):930–953.

DiMatteo, M. R., Taranta, A., Friedman, H., and Prince, L. (1980). Predicting patient satisfaction from physicians' nonverbal communication skills. *Med Care*, 18(4):376–387.

Finkelstein, A. (1991). A (neat) alphabet of requirements engineering issues. In van Lamsweerde, A. and Fuggetta, A., editors, *ESEC '91, 3rd European Software Engineering Conference, Milan, Italy, October 21-24, 1991, Proceedings*, volume 550 of *Lecture Notes in Computer Science*, pages 489–491. Springer.

Finkelstein, A. (2013). The next 10 years: the shape of software to come and what if means for software engineering. In Castro, J., Alencar, F. M. R., Lucena, M., and Filho, G. A. C., editors, *Proceedings of Requirements Engineering@Brazil 2013, Rio de Janeiro, Brazil, July 16, 2013*, volume 1005 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Fitzgerald, B. (2012). Software crisis 2.0. *Computer*, 45:89–91.

Gasevic, D., Kaviani, N., and Milanovic, M. (2009). Ontologies and software engineering. In *Handbook on Ontologies*.

Greenspan, Sol J.and Mylopoulos, J. and Borgida, A. (1982). Capturing more world knowledge in the requirements specification. In Ohno, Y., Basili, V. R., Enomoto, H., Kobayashi, K., and Yeh, R. T., editors, *Proceedings, 6th International Conference on Software Engineering, Tokyo, Japan, September 13-16, 1982*, pages 225–235. IEEE Computer Society.

Guarino, N., Guizzardi, G., and Mylopoulos, J. (2019). On the philosophical foundations of conceptual models. In Dahanayake, A., Huiskonen, J., Kiyoki, Y., Thalheim, B., Jaakkola, H., and Yoshida, N., editors, *Information Modelling and Knowledge Bases XXXI - Proceedings of the 29th International Conference on Information Modelling and Knowledge Bases, EJC 2019, Lappeenranta, Finland, June 3-7 2019*, volume 321 of *Frontiers in Artificial Intelligence and Applications*, pages 1–15. IOS Press.

Hadar, I., Zamansky, A., and Berry, D. M. (2019). The inconsistency between theory and practice in managing inconsistency in requirements engineering. *Empirical Software Engineering*, 24(6):3972–4005.

Hall, E. T., Birdwhistell, R. L., Bock, B., Bohannan, P., Diebold, A. R., Durbin, M., Edmonson, M. S., Fischer, J. L., Hymes, D., Kimball, S. T., La Barre, W., Lynch, F., S., J., McClellan, J. E., Marshall, D. S., Milner, G. B., Sarles, H. B., Trager, G. L., and Vayda, A. P. (1968). Proxemics [and Comments and Replies]. *Current Anthropology*, 9:83–108.

Hans, A. and Hans, E. (2015). Kinesics, Haptics and Proxemics: Aspects of Non -Verbal Communication. *IOSR Journal Of Humanities And Social Science (IOSR-JHSS)*, 20:47–52.

Ivanov, V., Rogers, A., Succi, G., Yi, J., and Zorin, V. (2017). What Do Software Engineers Care About? Gaps Between Research And Practice. In *Proceedings of the 2017 ACM SIGSOFT Symposium on the Foundations of Software Engineering (ESEC/FSE 2017)*, pages 890–895, Paderborn, Germany. ACM.

Kendon, A. (1967). Some functions of gaze-direction in social interaction. *Acta Psychologica*, 26:22–63.

Lasswell, H. D. (1948). The structure and function of communication in society. *In L. Bryson (Ed.), The communication of ideas*, pages 37–51.

Mooney, A. (2018). 6 problems that delay software projects.

Moote, M. A., Mcclaran, M. P., and Chickering, D. K. (1997). RESEARCH: Theory in Practice: Applying Participatory Democracy Theory to Public Land Planning. *Environmental Management*, 21:877–889.

Patton, W. and McMahon, M. (2014). *Career Development and Systems Theory: Connecting Theory and Practice*. Sense Publishers, 3 edition edition.

Pressman, R. S. and Maxim, B. R. (2014). *Software Engineering 8th ed.* McGraw-Hill Education.

Rice, R. W. and Perry, W. E. (2011). *Testing Dirty Systems*. CreateSpace Independent Publishing Platform.

Ruparelia, N. B. (2010). Software development lifecycle models. *ACM SIGSOFT Software Engineering Notes*, 35:8–13.

Salzmann, Z. (1998). *Language, culture and society: An introduction to linguistic anthropology (2nd ed.)* . Nashville TN: Westview.

Science Committee, N. (1968). Report on a conference sponsored by the nato science committee garmisch, germany, 7th to 11th october 1968.

Simon, H. A. (1969). *The Sciences of the Artificial*. London: MIT Press.

Standish Group (2014). CHAOS – The Standish Group Report.

Standish Group (2015). Chaos report 2015 by standish group.

Von Bertalanffy, L. (1972). Zu einer allgemeinen Systemlehre. In *Organisation als System*, pages 31–45. Springer.

Wagner, G. and Altenberg, L. (1996). Complex adaptations and the evolution of evolvability. *Evolution*, 50:967–976.

Wan, J. and R., W. (2010). Empirical research on critical success factors of agile software process improvement. *J. Software Engineering & Applications*, 3:1131–1140.

Wirth, N. (1995). A plea for lean software. *Computer*, 28:64–68.