Reflections on the Need for Disambiguation of Terminology for Software Process Improvement

Elli Georgiadou(✉)

Middlesex University, London, UK
e.georgiadou@mdx.ac.uk

Abstract. Researchers often invent new terms or borrow terms from other domains which they modify or reinterpret by attaching novel meanings to suit the context of their investigation. In established disciplines, such as traditional engineering, physics, and mathematics, there is no ambiguity in the relevant terminology as terms as well as their relationships are expressed formally, mostly using mathematical equations. In Software Engineering the fact that there is a need for a glossary or lists of definitions at the end of every book and within nearly all research papers, is evidence that the discipline is not yet mature enough even after 50 years from the coining of the term Software Engineering. Refinements and re-interpretations of existing terms, combined with new terms introduced with every innovation, every new technology or tool, by both researchers and practitioners, result in ambiguities which need to be addressed. As change is inevitable there needs to be a mechanism through which ambiguities can be identified and addressed. This paper outlines major advances in Terminology Management for Software Process Improvement and reflects on the constant need for terminology disambiguation. It proposes a bottom-up pragmatic approach for using ontologies and for integrating research and practice. It also proposes a framework based on SPI Manifesto values and principles for the disambiguation of terms for continuous process improvement and, thus, for enhancing value and for catering for change which is inevitable.

Keywords: Software measurement · Disambiguation · Terminology Communication · SPI values & principles · Framework

1 Introduction

"Speaking the same language" is a common phrase which does not necessarily refer to a natural language like English, German, Chinese… but more significantly, it means common understanding and agreeing with each other. Lack of ambiguity is a fundamental requirement for common understanding and communication. Understanding one another takes place "because there exists a code, a sort of inner competence between you and me and there exist possible messages, performed as utterances and interpretable as a set of propositions" [1].

In natural languages we use words as synonyms and thus we use them interchangeably but mostly when we intent to place particular emphasis on what we are

X. Larrucea et al. (Eds.): EuroSPI 2018, CCIS 896, pp. 577–589, 2018.
https://doi.org/10.1007/978-3-319-97925-0_49

trying to communicate. For example an item can be referred to as an object, a thing, an artefact and so on.

In pedagogy for example, the recent explosion of technology based Distance Education (DE) has resulted in the creation of new terms which, however, are often used interchangeably as synonyms. Bozkurt et al. [2] in their examination of 633 scholarly articles (covering 5 years from 2009 to 2013) they reported that at least 12 terms are being used by scholars to describe DE.

An example of erroneous use of terms as synonyms from the Information Systems and Software Engineering domains is that of method and methodology. The use of these terms as synonyms has been perpetuated for over 40 years. Evidence of this issue can be found in many articles and acclaimed textbooks such as Avison and Fitzgerald [3] and Jayaratna [4].

Howell, in her book "Introduction to the Philosophy of Methodology" [5] clarifies that "a methodology is the rationale for the research approach, and the lens through which the analysis occurs. Said another way, a methodology describes the general research strategy that outlines the way in which research is to be undertaken…A method is simply a tool used to answer your research questions—how, in short, you will go about collecting your data. Examples of research methods include: Contextual inquiry, Interview, Survey etc. The methodology should impact which method(s) for a research endeavour are selected in order to generate the compelling data."

Another term with many and varied meanings, in everyday life but also in Software Engineering and Information Systems, is the term model which can be a representation, a pattern, a prototype, a template, a blueprint, an exemplar, a framework, a paradigm and so on. Each model represents an item or a process from a specific perspective. In Software Engineering the term model could denote many different things including lifecycle development model, a process model, a reference model, a quality model, a framework, a meta-framework and so on.

According to Barn [6] "conceptual modelling provides a mechanism by which a shared understanding between business domain specialists and IT specialists positively enhances the alignment of business and IT goals leading to improved quality of IT Solutions". Thomas [7] cited in [6] argued that "no uniform grasp of the term reference model exists". This confusion partially arises out of the tendency to declare Application information models and/or enterprise Information models as "reference models". Barn et al. [8] following a review of existing reference models synthesised a definition: "A reference model is based on a small number of unifying concepts and is an abstraction of the key concepts, their relationships, and their interfaces both to each other and to the external environment. A reference model may be used as a basis for education and for explaining standards and methods to a non- specialist and can be viewed as a framework for comparing architectures and operations of existing and future systems."

In the disciplines of Software Engineering and Information Systems often many different terms are used to define the same concept. Conversely, the same term may be used to define several concepts. For example, numerous definitions of Software Quality have been proposed. Kitchenham and Pfleeger [9] identified five different perspectives of quality as follows:

- The transcendental perspective deals with the metaphysical aspect of quality;
- The user perspective is concerned with the appropriateness of the product for a given context of use;
- The manufacturing perspective represents quality as conformance to requirements;
- The product perspective implies that quality can be evaluated by measuring the inherent characteristics of the product;
- The value-based perspective of quality recognises that the different perspectives of quality may have a different importance, or value, to various stakeholders.

In Siakas et al. [10] we produced an alphabet of software quality in order to emphasise the fact that numerous terms are used by the community to describe characteristics and sub-characteristics of quality which were (and still are) being used by the community. In this paper we did not attempt to show interrelationships of the terms but produced a high level taxonomy of three classes of attributes namely those of primary interest to developers, sponsors, or users. In all 24 terms were found in the literature on the main software quality models (at the time) i.e. McCall, Boehm, Dromey, SADT, and ISO 9126 (Y proved impossible to include) and Zoticality ζ (Z was coined). In the process it was established that although there was considerable overlap between the models, many terms were used to describe the same concept and there were differences in terminology between models.

Georgiadou [11] in “Navigating the Labyrinth of Software ‘Re’ Words” provided definitions largely based on IEEE and differentiated between reuse, restructuring, re-engineering, retro-engineering, reverse engineering and refactoring. By identifying how these terms relate to each other, and specified what quality improvements can be gained through practicing each of these processes. In addition the benefits of the proposed disambiguation of these terms and potential problems were specified.

Kitchenham [12] encapsulated many of the issues relating to the understanding, definition, and measurement of Software Quality. She argued that “Quality is a complex concept that means different things to different individuals. It can be highly context dependent. This means that there can never be any simple measure of quality that will be accepted by everyone. If you are interested in assessing or improving quality in your organisation, you must ensure that you define what aspect of quality you are interested in and how you are going to measure it. In fact, if you define quality in a measurable way, it is usually easier for other people to understand your viewpoint”.

Abran et al. [13] noted that “from the metrology perspective suggests that the field of software measurement has not yet been fully addressed by current research, and that much work remains to be done to support Software Engineering as an engineering discipline based on quantitative data and adequate measurement methods meeting the classic set of criteria for measuring instruments as described by the metrology body of knowledge in large use in the engineering disciplines”.

Despite the many advances achieved by the Software Engineering community, it continues to be reported by both researchers and practitioners such as Rout [14]; Gilb [15]; Garcia et al. [16]; Navigli [17]; Prokofyev et al. [18], Clarke et al. [19]; Jacobson et al. [20]; Kirsch and Sauberer [22] Sauberer et al. [22], Clarke et al. [37] that there is a need to standardise terminology in order to facilitate communication, understanding, and team work, and thus achieve improvements in process and project management.

2 Terminology Management: Vocabularies, Taxonomies, and Ontologies

The significance of terminology management becomes evident when we look at the concerns of and debates within the Software Engineering Community (practitioners and academics) expressed in the literature. Garcia-Penalvo et al. [16] state that "Terminology is vital to the functioning of all sciences, it is concerned with designations in all other subject fields, and it is closely related to a number of specific disciplines, as already pointed out by its most distinguished modern protagonist, Eugen Wuster who called it an interdisciplinary field of study, relating linguistic, logic, ontology and information science with the various subject fields."

Taking an example from the field of Medicine, Hu et al. [24]) argue that "generalists typically attach philosophical sophistication to their approach, in supposed contrast to the narrow remit chosen by the application-bound knowledge engineers, we would like to indicate that the latter practice can often reflect a multi-faceted rationale, nuanced by the requirements of the domain." One can imagine how significant it is for doctors to speak the same language with other.

Rout [14] argues that semantic technologies are based on ontologies…Ontology formalizes knowledge meaning and facilitates the search for contents and information…The main objective of ontologies is to establish ontological agreements, which serve as the basis for communication between either human or software agents, hence, reducing language ambiguity and knowledge differences between agents, which may lead to errors, misunderstandings and inefficiencies.

Navigli [17] in a comprehensive survey concluded: "Although some contrasting works have been published on the topic, (of Word Sense Disambiguation) no conclusive result has been reported in favour or against the use of sense inventories and their granularity in applications. Unsupervised approaches might prove successful, showing that we do not need to rely on predefined lists of senses. However, it might be as well that the use of sense inventories with a certain granularity (not too fine-grained nor trivially coarse) allow knowledge-rich and supervised methods to provide a decisive contribution".

Sauberer et al. [22] in their seminal work "proposed an extension of terminology change management procedure and the development of an e-learning course which offers a user-friendly and sound introduction to basic principles and methods of terminology management."

Jacobson et al. [20, 25] believe that Software Engineering is gravely hampered today by immature practices. Specific problems include: the prevalence of fads more typical of fashion industry than of an engineering discipline, the lack of a sound, widely accepted theoretical basis, the huge number of methods and method variants, with differences little understood and artificially magnified, the lack of credible experimental evaluation and validation, the split between industry practice and academic research. The signatories and founders of SEMAT support a process to re-found software engineering based on a solid theory, proven principles and best practices that: includes a kernel of widely-agreed elements, is extensible for specific uses, address both technology and people issues which are supported by industry, academia, researchers and users, and also support extension in the face of changing requirements and technology. Table 1 presents a chronology of seminal contributions to the terminology debate, management and practice.

Table 1. The chronology of seminal contributions in the terminology debate, management and practice

Ref	Researcher (s)	Focus	Major contribution
[14]	Rout (1999)	Established that there are duplications and discrepancies in definitions of terms in various standards	Preliminary evaluation of consistency and terminology in the group of software engineering standards developed by ISO/IEC JTC1/SC7
[16]	Garcia et al. (2006)	Vocabulary conflicts and inconsistencies Between various standards	Basic software measurement ontology aligned with metrology
[15]	Gilb (2006)	Planguage	Definition of concepts – not terms
[26]	Gómez-Pérez et al. (2006)	Ontologies	Ontologies include a vocabulary of terms, and some specification of their meaning
[17]	Navigli (2007)	Word sense disambiguation	Automatic identification of correct sense for scientific literature, Formal taxonomies
[21]	Karsch and Sauberer (2011)	Terminological precision	Key Factor in Product Usability and Safety, in Design, User Experience and Usability, Theory, Methods, Tools and Practice architecture for
[27]	Henderson-Sellers (2011)	Metamodels and ontologies	Application of ISO/IEC architecture for ontology life-cycle management
[16]	Garcia-Penalvo et al. (2012)	Ontology modelling tool	Semantic technologies are based on ontologies Ontology formalizes knowledge meaning and facilitates the search for contents and information
[25]	Jacobson et al. (2013)	A thinking framework	SEMAT Kernel Software Engineering Method and Theory
[26]	Jacobson et al. (2013)	SEMAT Kernel	Practice-independent foundation for the definition of software methods, the kernel also has the power to completely transform the ways that methods are defined and practices are shared

(*continued*)

Table 1. (*continued*)

Ref	Researcher (s)	Focus	Major contribution
			SEMAT is Actionable, Extensible, Practical
[18]	Prokofyev et al. (2013)	Word sense disambiguation (WSD: the ability to identify the meaning of words in context in a computational Manner	Ontology based disambiguation
[19]	Clarke et al. (2016)	Problems with terminology diversity	Sustained effort by multiple disciplines, including terminology expertise, software development specialist, knowledge management know-how and computational linguistics Towards a canonical software development process and roles ontology
[22]	Sauberer et al. (2017)	Content development of terminology management	Extension of terminology change management procedure The development of an e-learning course which offers a user-friendly and sound introduction to basic principles and methods of terminology management

3 Communication Through Consistent Terminology and Ontologies

Misuse and polymorphism of terms hinders effective communication, resulting in miscommunication and even conflict and failure. The CHAOS Reports identify miscommunication between business and IT personnel as one of the main reasons for systems failures is [28].

The use of a common terminology among all stakeholders enables effective knowledge sharing and lays the foundation for process improvement and success of projects. One of the Principles of the SPI Manifesto is to "Ensure all parties understand and agree on process" [http://2012.eurospi.net/index.php?option=com_content&view=article&id=32&Itemid=33].

Consistent terminology minimises miscommunication and develops common understanding. Terminology involves the terms themselves, their meanings, but also and the relationships between them. The adoption of consistent terminology helps avoid each researcher inventing a language of their own from scratch. Pfitzmann and Hansen [29] asserted: "Of course, each paper will need additional vocabulary, which might be added consistently to the terms defined here". Thus by showing relationships between terms a list or glossary of terms is converted to a consistent terminology.

Quality Models such as the McCall, Boehm, Dromey, ISO9126 etc. reviewed in Côté et al. [30] are expressed in textual form but also in high level diagrammatic representations with characteristics and sub-characteristics where the decomposition of each characteristic indicates the relationship ‘belongs to’. Côté et al.’s review showed commonalities across these Quality Models but also some inconsistencies and ambiguities in the classification and meaning of several terms. For example Correctness is decomposed into Traceability, Completeness and Consistency whilst in McCall Correctness is analysed into Functionality and Reliability by Dromey.

Masolo and Borgo [31] emphasize that “we need to find a common ground, as different researchers often rely on definitions of quality referring to quite different things: adherence to requirements, fitness for a particular purpose, user satisfaction etc. We believe that it can be done based on the recent efforts of establishing the notion of software quality and its usage in terms of the formal ontology”.

Calero et al. [32] in an extensive review of the literature concluded that ontologies in Software Engineering and Technology aim to share knowledge of the problem domain and use a common terminology among all interested parties (not only researchers), and to filter the knowledge when defining models and metamodels.

Gómez-Pérez et al. [26] argue that “ontologies aim to capture consensual knowledge in a generic way, and that they may be reused and shared across software applications and by groups of people. Ontologies include a vocabulary of terms, and some specification of their meaning. This includes definitions and an indication of how concepts are interrelated, which collectively impose a structure on the domain and constrain the possible interpretations of terms.”

Jacobson et al. [25] explain that kernel allows you to add practices, such as user stories, use cases, component-based development, architecture, pair-programming, daily stand-up meetings, self-organizing teams, and so on to build the methods you need. Mahon [36] emphasises that practitioners can use the SEMAT Kernel and particularly the pre-prepared cards to guide practitioners so that they can carry out their processes efficiently and effectively.

4 The Role of Standards in Reducing of Ambiguity

A standard is defined by Gonzalez-Perez et al. [21] as a “document, established by consensus and approved by a recognized body, that provides, for common and repeated use, rules, guidelines or characteristics for activities or their results, aimed at the achievement of the optimum degree of order in a given context”. Standards are needed in order to avoid idiosyncrasy, to ensure repeatability, to reach consensus, and reassure customers. Most products have safety standards but process standards are more rare, and are also much more difficult to enforce. Standardising a process can ensure that no steps are skipped. Also a standardized process which is understood and accepted/institutionalised can improve quality (productivity and reliability).

Gonzalez-Perez et al. [33] refer to the need for various ISO International Standards to be consistent with each other in terms of terminology, structure and semantics. They propose a potential harmonization across ISO software engineering standards (within Sub-Committee 7 (SC7) of ISO/IEC within Joint Technical Committee 1 (JTC1) by creating a core set of concepts and their relationships, effectively creating an abstract domain ontology for software engineering standards creation and utilization.

Rout [14] analysed a number of SC7 standards, documenting in which standards identified terms occurred together with their (disparate) definitions, only recently has harmonization become a crucial action point (rather than a vague concern) in SC7, especially under Special Working Group 5 (SWG5). Rout provided a table of Duplicated terms in SC7 Standards and related documents. He stated that his effort at the time was "…only to establish the necessity of resolving these differences for further improvement within the standards and processes in the software-engineering domain."

The INTERNATIONAL STANDARD ISO/IEC/IEEE 24765 First edition 2010-12-15 Systems and software engineering—Vocabulary was published by ISO JTC 1/SC 7, in 2010 [23]. This International Standard was prepared to collect and standardize terminology "The scope of each concept defined has been chosen to provide a definition that is suitable for general application. In those circumstances where a restricted application is concerned, a more specific definition might be needed. Terms have been excluded (a) if they were considered to be parochial to one group or organization; (b) if they were company proprietary or trademarked; (c) if they were multi-word terms whose meaning could be inferred from the definitions of the component words; and (d) if their meaning in the information technology (IT) field could be directly inferred from their common English meaning. Approximately two- thirds of the definitions in this International Standard are new since IEEE Std 610.12 was last updated in 1990, a reflection of the continued evolution in the field".

ISO/IEC TR 14143-3:2003, [34] Information Technology—Software measurement—Functional size measurement—Part 3: Verification of functional size measurement methods (term 3.8) defines the term repeatability (of results of measurements) i.e. closeness of the agreement between the results of successive measurements of the same measurand carried out under the same conditions of measurement. Repeatability may be expressed quantitatively in terms of the dispersion characteristics of the results. [International vocabulary of basic and general terms in metrology, 1993, definition 3.6].

5 Universal Concepts, Research Ambitions and Pragmatic Customisation

Researchers often start their investigations with a top-down, wide ranging ambition spanning a large knowledge area even involving universal concepts, ideas, themes, and principles that are found and can be proven within, between, and across various subject areas and disciplines. As the researchers gradually gain insights and understanding of the enormity and complexity of their originally ambitious scope and purpose, they narrow down their investigation focussing on more specific problems. For example Software Quality Models some of which were mentioned earlier in this paper aim to provide a universal framework for all types of software systems. However, it is only when they pragmatically consider the methods and techniques used for various categories, types and complexity of a more narrow selection of systems, they gain further insights and they produce bottom-up tangible solutions.

During their journey from the general to the specific and then to the general they often generate new knowledge expressed in novel interpretations of existing terms or they even generate new terms in order to exemplify their research contribution, the originality of their work. Continuous change and continuous improvements, innovations and changes in technologies bring their own new terms (which may be synonyms to existing terms) to add to the already highly populated vocabulary of Software Engineering. This in turn generates ambiguity.

Practitioners are normally focussed on in-house projects and challenges. Their own experiential knowledge is often very useful for the theoreticians especially when they form part of the same team integrating theory and practice.

For continuous terminology improvement and process improvement too, we need to always revisit our initial research question to ensure validation and verification of our findings. As T.S. Eliot said "We shall not cease from exploration. And the end of all our exploring will be to arrive where we started and know the place for the first time". T.S. Eliot, Little Gidding (1888–1965).

Šmite et al. [35] observed that "there is no unified empirically based glossary or reference model established in the field, which contains a clear description of the various terms that are used to describe sourcing strategies, or a taxonomy that would explicitly define the relationships between the different terms. They proposed a taxonomy for Global Software Engineering by categorizing the elected terms based on generalization-specialization relationships and illustrate how the taxonomy can be used to categorize and map existing knowledge. The contribution targets future researchers, who will publish or synthesize further empirical work and practitioners, who are interested in published empirical cases. Therefore this work is expected to make a contribution to the future development of research in the GSE field, and alleviate Understandability and transferability of existing and future knowledge into practice.

6 A Framework for Achieving Disambiguation of Terms

Following this investigation, we propose a framework (depicted in Fig. 1) for launching, carrying out and implementing the terminology disambiguation process. The central node (Number 7) forms the core of the framework. It is the on-going sub-process of reviewing all the works coming from all other stages and all committees, sub-committees, standards bodies and so on. This is a continuous and cyclic monitoring and co-ordinating sub-process. The bidirectional arrows emphasise the need of change of information, updating all stakeholders and responding to external changes.

Questions to be posed and answered at this stage include:

Which terms and combinations of terms are used to characterise different elements, factors, actors, actions?

- Which terms are erroneous and Why?
- Which terms are deficient?
- Which terms are new? Are they related to existing terms?
- What are the unambiguous, correct terms, their definitions and their relationship to other terms.

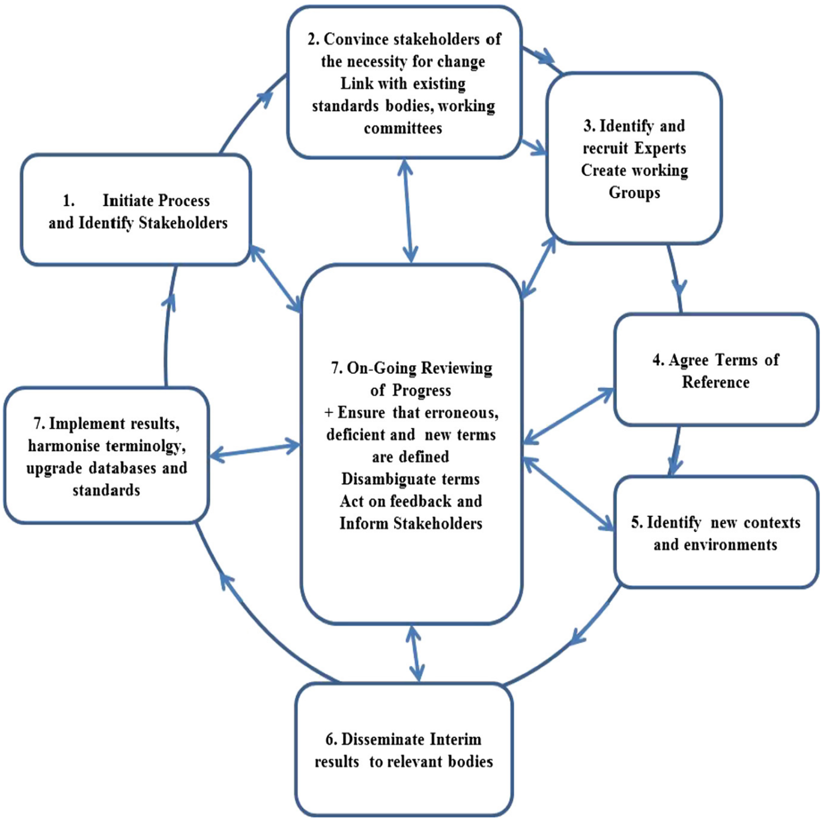
2. Convince stakeholders of the necessity for change Link with existing standards bodies, working committees
3. Identify and recruit Experts Create working Groups
1. Initiate Process and Identify Stakeholders
7. On-Going Reviewing of Progress + Ensure that erroneous, deficient and new terms are defined Disambiguate terms Act on feedback and Inform Stakeholders
4. Agree Terms of Reference
7. Implement results, harmonise terminolgy, upgrade databases and standards
5. Identify new contexts and environments
6. Disseminate Interim results to relevant bodies

Fig. 1. A framework for the effective disambiguation of terms

Throughout the whole process pertinent questions need to be aked and answered. For example: “Who are the stakeholders? “Some of these stakeholders will be responsible for finanancing the intitiative, some will be carrying out the examination of existing terms, etc. hence at stage 2 we need to convince all stakeholders that change is necessary so the question “Why change “must be posed and answered. Terms of reference (stage 3) include questions like “What is the budget?”, “What is the time-frame for the completion of the tasks”, “What are the roles and responsibilities?”, “Who are the people undertaking these roles”.

There are distinct roles for carrying out administrative and management tasks. Domain experts are involved in the identification of inconsistencies, deficiencies, and errors in terms existing terms. Terms and their relationship to other terms may be accepted, rejected, modified, refined, and incoprorated in existing databases. If experts disagree and cannot reconcile their differences in opinion terms may be rejected or revisited for further elaboration.

7 Conclusion

7.1 Value Gains from the Elimination of Ambiguities in Terminology

Ambiguities in terminology can be addressed and minimised even eliminated through the use of ontologies which are context specific. This is the reason that there exist Ontologies for Requirements, for Risk Management, for Project Management etc. Ontologies reduce conceptual and terminological ambiguity, as they provide us with a framework for unification which in turn facilitates communication and knowledge sharing among diverse viewpoints, contexts, cultures etc.

The latest important contribution comes from Ralph [38] in his paper “Toward Methodological Guidelines for Process Theories and Taxonomies in Software Engineering”. He provides an incisive analysis of the issues, “clarifies the nature and functions of process theories and taxonomies in software engineering research, and synthesizes methodological guidelines for their generation and evaluation”. He is challenging us with areas of future research.

The systems and software engineering disciplines are continuing to mature while information technology advances. New technological innovations and methodological thinking reflect the rapid change. The multi-disciplinarity of Software and Systems Engineering brings immense complexity and challenges to academics and practitioners alike. New terms are being generated and new meanings are being adopted for existing terms (INTERNATIONAL STANDARD ISO/IEC/IEEE 24765).

Process improvement can be achieved and value gains can be achieved for the benefit of the organisation, the project, the individuals involved, and the whole Software Engineering community through avoiding miscommunication, confusion and even conflict. Only when agreed and accepted standards of incorporating all the revised, refined or totally new terms are systematically incorporated in a consistent and unambiguous terminology.

7.2 Future Work and Acknowledgements

Future work will involve the refinement and validation of the proposed framework, and development of technical infrastructure which will enable further automation of the process of term disambiguation.

Acknowledgements and thanks are due to the anonymous reviewers for their thorough reviews, critical comments, expert advice, and useful suggestions for improving the original submission.

References

1. Eco, U.: Social life as a sign system. In: Robey, D. (ed.), Structuralism: An Introduction, Clarendon, Oxford (1973)
2. Bozkurt, A., et al.: Trends in Distance Education Research: A Content Analysis of Journals 2009–2013 (2015)
3. Avison, D., Fitzgerald, G.: Information Systems Development: Methodologies, Techniques and Tools. Graw-Hill (2003)

4. Jayaratna, N.: Understanding and Evaluating Methodologies, NIMSAD: A Systemic Approach, McGraw-Hill (1996)
5. Howell, K.E.: An Introduction to the Philosophy of Methodology. Sage Publications, London (2013)
6. Barn, B.S.: On the evaluation of reference models for software engineering practice. In: Proceedings of the 2nd India Software Engineering Conference, ISEC 2009, pp. 111–116 (2009)
7. Thomas, O.: Understanding the term reference model in information systems research: history, literature analysis and explanation. In: Bussler, Christoph J., Haller, A. (eds.) BPM 2005. LNCS, vol. 3812, pp. 484–496. Springer, Heidelberg (2006). https://doi.org/10.1007/11678564_45
8. Barn, B.S., Dexter, H., Oussena, S., Petch, J.: An approach to creating reference models for SOA from multiple processes. In: IADIS Conference on Applied Computing, Spain (2006)
9. Kitchenham, B., Pfleeger, S.L.: Software quality: the elusive target. IEEE Softw. **13**, 12–21 (1996)
10. Siakas, K.V., Berki, E., Georgiadou, E., Sadler, C.: The complete alphabet of quality software systems: conflicts and compromises. In: 7th World Congress on Total Quality & Qualex 97, New Delhi, India, 17–19 February (1997)
11. Georgiadou, E.: Navigating the labyrinth of software 're' words. In: Dawson, R., Ross, M., Staple, G. (eds.) Proceedings of the 17th International Conference Software Quality Management (SQM), Software Quality in the 21st Century. BCS (2009)
12. Kitchenham, B.: Software Metrics, Measurement for Software Process Improvement. NCC/Blackwell, Oxford (1996)
13. Abran, A., Sellami, A., Suryn, W.: Metrology, measurement and metrics in software engineering. In: Proceedings of the 5th International Workshop on Enterprise Networking and Computing in Healthcare Industry (IEEE Cat. No. 03EX717) (2003)
14. Rout, T.P.: Consistency and conflict in terminology in software engineering standards. In: Proceeding of the Fourth IEEE International Symposium and Forum on Software Engineering Standards (ISESS 1999), pp. 67–74. IEEE Computer Society Press, Los Alamitos (1999)
15. Gilb, T.: Competitive Engineering: A Handbook For Systems Engineering, Requirements Engineering, and Software Engineering Using Planguage. Elsevier Butterworth-Heineman, Oxford (2005)
16. Garcia-Penalvo, F.J., Colomo-Palacios, R., Garcia, J.: Towards an ontology modelling tool. A validation in software engineering scenarios. Expert Syst. Appl. **39**(13), 11468–11478 (2012)
17. Navigli, R.: Word sense disambiguation: a survey. ACM Comput. Surv. **41**(2), 10:1–10:69 (2009)
18. Prokofyev, R., Demartini, G., Boyarsky, A., Ruchayskiy, O., Cudré-Mauroux, P.: Ontology-based word sense disambiguation for scientific literature. In: Serdyukov, P., et al. (eds.) ECIR 2013. LNCS, vol. 7814, pp. 594–605. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36973-5_50
19. Clarke, P.M., et al.: Refactoring software development process terminology through the use of ontology. In: Kreiner, C., O'Connor, Rory V., Poth, A., Messnarz, R. (eds.) EuroSPI 2016. CCIS, vol. 633, pp. 47–57. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-44817-6_4. ISBN 978-3-319-44817-6
20. Jacobson, I., Pan-Wei, N., McMahon, P., Spence, I., Lidman, S.: The Essence of Software Engineering—Applying the SEMAT Kernel. Addison-Wesley, Boston (2013). Forthcoming in January 2013 but available in a prepublication version on safaribooksonline.com

21. Karsch, B.I., Sauberer, G.: terminological precision - a key factor in product usability and safety. In: Marcus, A. (ed.) DUXU 2011. LNCS, vol. 6769, pp. 138–147. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-21675-6_16
22. Sauberer, G., Villar, B.N., Dreßler, J.R., Schmitz, K.-D., Clarke, P.M., O'Connor, R.V.: Do we speak the same language? Terminology strategies for (software) engineering environments based on the Elcat model - innovative terminology e-learning for the automotive industry. In: Stolfa, J., Stolfa, S., O'Connor, Rory V., Messnarz, R. (eds.) EuroSPI 2017. CCIS, vol. 748, pp. 653–666. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-64218-5_54. ISBN 978-3-319-64218-5
23. ISO/IEC/ IEEE 24765, ISO International Standard ISO/IEC/ IEEE 24765 First edition 2010-12-15 Systems and software engineering—vocabulary was published by ISO JTC 1/SC 7
24. Hu, B., Dasmahapatra, S., Dupplaw, D., Lewis, P., Shadbolt, N.: Reflections on a medical ontology. Int. J. Hum.-Comput. Stud. **65**(7), 569–582 (2007)
25. Jacobson, I., Ng, P.-W., Spence, I.: Enough of process—let's do practices. J. Object Technol. **6**(6), 41–67 (2007)
26. Gómez-Pérez, A., Fernandez-Lopez, M., Corcho, O.: Ontological Engineering: With Examples from the Areas of Knowledge Management, e-Commerce and the Semantic Web. Springer, London (2006). https://doi.org/10.1007/b97353
27. Henderson-Sellers, B., Gonzalez-Perez, C., McBride, T., Low, G.: An ontology for ISO software engineering standards: 1) creating the infrastructure. Comput. Stand. Interfaces **36** (3), 563–576 (2014)
28. Standish Group: The CHAOS Report. The Standish Group International (1994). http://www.standishgroup.com/sample_research/chaos_1994_1.php
29. Pfitzmann, A., Hansen, M.: A terminology for talking about privacy by data minimization: anonymity, unlinkability, unobservability, pseudonymity, and identity management – version v0.34. Technical report, TU Dresden and ULD Kiel (2011)
30. Côté, M.-A., Suryn, W., Georgiadou, E.: In search for a widely applicable and accepted software quality model for software quality engineering. Softw. Qual. J. **15**(4), 401–416 (2007)
31. Masolo, C., Borgo, S.: Qualities in formal ontology. In: Hitzler, P., Lutz, C., Stumme, G. (eds.) Proceedings of the Workshop on Foundational Aspects of Ontologies (FOnt 2005), pp. 2–16 (2005)
32. Calero, C., Ruiz, F., Piattini, M. (eds.): Ontologies for Software Engineering and Software Technology. Springer, Heidelberg (2006). https://doi.org/10.1007/3-540-34518-3
33. Gonzalez-Perez, C., Henderson-Sellers, B., McBride, T., Low, G.C., Larrucea, X.: An ontology for ISO software engineering standards, 2) proof of concept and application. Int. J. Hum.-Comput. Stud. **65**(2007), 569–582 (2007)
34. ISO/IEC TR 14143-3:2003: [23] Information technology—Software measurement—Functional size measurement—Part 3: Verification of functional size measurement methods
35. Šmite, D.: An empirically based terminology and taxonomy for global software engineering. Empir. Softw. Eng. **19**(1), 105 (2014). ISSN 1382-3256
36. Mahon: The Essence of Software Engineering - Applying the SEMAT Kernel. With Pan Wei Ng, Paul Mc Mahon, Ian Spence and Svante Lidman. Addison-Wesley (2013)
37. Clarke, P., et al.: An Investigation of Software Development Process Terminology. In: Clarke, Paul M., O'Connor, Rory V., Rout, T., Dorling, A. (eds.) SPICE 2016. CCIS, vol. 609, pp. 351–361. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-38980-6_25
38. Ralph, P.: Toward methodological guidelines for process theories and taxonomies in software engineering. IEEE Trans. Softw. Eng. (2018). https://doi.org/10.1109/TSE.2018.2796554