Position Statement for SEMAT

Brian Henderson-Sellers, PhD, DSc, FIEAust. FIMA, MASCE

SEMAT's goal is to "refound software engineering based on a solid theory, proven principles and best practices"

[Since we are discussing theory, I have included many citations to the peer-reviewed literature to illustrate what is already available. I apologize that many are self-citations but this is indicative that my work has been solidly in the SEMAT area over the last two decades.]

The first problems to solve are essentially

- What makes a topic area into a professional *engineering discipline*?
- What is the appropriate underpinning theory(s)?

Having established the scope and an appropriate suite of theories, a kernel can then be constructed that embodies this theoretical basis by means of conceptual models, principles and practices.

Engineering is the application of scientific knowledge to problem solving in the real world. An engineering discipline does not have a theory *per se* but relies on theories developed in other, basic scientific disciplines. For example, civil engineering uses the theory of materials and the mathematics of stress analysis; mining engineering the knowledge gained in chemistry. Engineering itself develops problem solving methods in a technological adjunct to the area of its supporting science. Problem solving must therefore be consistent with the underlying theory of this basic science but also be shown to be practically useful, reliable and of high quality. This introduces the elements of empirical research to build up an extensive experience base and of quantification into the basic notion of an engineering discipline. Indeed, without data (empirical evidence) one can never have a professional discipline. The American Society of Civil Engineers' definition of (civil) engineering encompasses codes, standards, a body of knowledge and ethics; SWEBOK's definition of software engineering involves it being "systematic, disciplined and quantifiable"; and Gilb has recently proposed three key areas as being "Principles, Concepts and Measures". Thus "Quantify and Codify" could be an engineer's slogan.

What is an appropriate underpinning theory[1] for software engineering? To identify the concepts required at this "basic" level (1. Basics in Figure 1), we must aim for a highly abstract set of concepts that could be said to underpin software engineering. I propose that we go back to real basics in terms of ontologies (which are closely associated with metamodels:[4]) and measurement (e.g. many metrics authors such as Zuse, Fenton, Constantine, Kitchenham, Jeffery, Verner, Card, Briand).

Metamodels and ontologies have been developed over the last decade both in parallel and synergistically [2,5,6,15]. For example, Figure 2, extracted from ISO/IEC 24744 International Standard [14] and consistent with other approaches such as OPF, SPEM and that of Kruchten [16] (who offers eight key concepts), shows a core metamodel that underpins software engineering development *methodologies*. Typically, these core elements, which are of necessity orthogonal, are aligned with the question words of the Zachman framework [26]. Whatever the actual final set of agreed concepts (a number probably between 5 and 8), these define the basic ontology, which consequently underpins an agreed terminology. Synonyms should be allowed (and documented to ensure standard usage) so long as it is clear that a Task in Country or Organization A is *exactly* semantically equivalent to, say, an Activity in Country B

In the literature there are metamodels (and associated ontologies) for methodologies [14], for metrics [20], for ontologies [21], for organization structure [22] and business process definition [19], for example. However, as noted in many, many emails to the OMG's OADTF discussion group, these are all effectively in non-communicating "silos". A critically useful result of adopting a single set of basic concepts is the opportunity for all metamodels to be made inter-consistent – which also permits the notion of situational metamodel engineering (SMME: [11]).

The Basics level must also contain the theory of measurement [24] – underpinning as it does software

[1] Almost a decade ago, Zendler [27] made such a proposal based on a researched history (back to 1976) of empirical software engineering.

metrics, and in particular quality metrics [28] – see also work by Briand and a simplified version that I used in my study of object-oriented metrics [8]**.** Level 1 Basics would combine measurement theory with a metrics metamodel – actual metrics to be developed later in Level 2 and validated in Level 3.

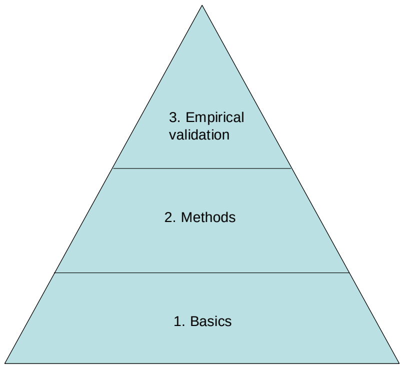

Figure 1 Three layers proposed for SEMAT

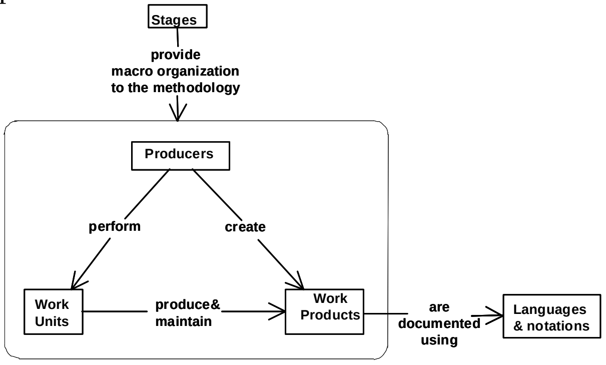

Figure 2 Five major metaclasses in ISO/IEC24744 (based on OPF and SPEM)

Thus, our Basics layer (Layer 1) has a reliance on logic and mathematics. Perhaps later, we might investigate more sophisticated formal (mathematical) underpinnings as was done for object technology and OO metrics by Whitmire [25].

A good engineering discipline, whilst embodying its theoretical underpinnings, must also learn from experience. A commonly quoted example is the new knowledge acquired as a result of the failure of the Tacoma Narrows Bridge when subjected to unexpected stress patterns. In designing the bridge, some theoretical knowledge had been ignored so that when the wind created a simple harmonic motion in the bridge, disaster ensued.

Once the contents of this "Basics" layer have been established, we can build upon them (labelled 2. Methods in Figure 1). In the methodological context embodied in the Draft Statement of Principles, the most obvious approach would be to use Situational Method Engineering [1,9,17,23] in which method fragments are created, each being conformant to one of the classes in the (Level 1) metamodel. These are stored in a repository or "methodbase" such as that of OPF (http://www.opfro.org and http://www.open.org.au), standardized along the lines of ISO/IEC 12207 [12] and (for capability assessment) ISO/IEC 15504 [13]. SWEBOK style KAs can then be mapped to elements in the methodology part of the metamodel (KAs 1-3, 8-9), to the metrics part(KAs 4-6) and to the organization component (KA7).

It is important to stress that in Level 1 and, especially in Level 2, we need an agreed semantics – this can be defined in the ontology (again allowing for synonyms).

Now, whether it is just Level 1 that is the "kernel" of the SEMAT documents on the website or whether the kernel is intended to include both Levels 1 and 2[2], it is clear that, using an SME approach, both levels 1 and 2 can readily be both minimalistic and extensible. Metamodels are easily extended [7], especially when based on the powertype approach of Odell [18] – see [10]; and the contents of a methodbase are easily extended. Indeed, extensibility is the whole rationale behind the SME approach.

Finally, the top layer (3. Empirical validation) is an evaluation of what works and what doesn't. Based on the knowledge of the methods layer, the SE community can start to make a serious attempt at providing good quality empirical evidence as to "good practice". Clearly we already have a lot of candidates for best practice endorsement – most are simply asserted to be so (with zero empirical validation). Now, we must instigate a rigorous evaluation process [3] before accepting their validity and before incorporating them into the body of knowledge. The SWEBOK contents are clearly able to be leveraged significantly in this endeavour.

At every stage, we need to involve standards organizations – primarily ISO with feed-ins from groups

[2] For instance, one position statement focusses on teams, another on metrics. Both would appear to be Level 2 rather than "basics".

like the OMG, IEEE/FIPA, W3C.

Finally, what is the *process by which SEMAT might achieve these goals*. I propose that we utilize our large number of volunteers in a hierarchical way. The basics, as I have proposed in Figure 2, probably with additions from Kruchten [16], identify at least four major strands viz. people, work products (including modelling languages), work units, temporal scheduling plus a fifth critical element: measurement. A senior SEMAT member is given overall responsibility for each area. Then within each of these 5 areas, individual topics are identified. Here, we could perhaps use the SWEBOK ten areas as a guide. Thus under work units, we might have focus areas of RE, design, testing etc. For each focus group, a team is appointed to gather all the published literature, stressing both theoretical contributions equally with good quality industry data but avoiding polemics and unsubstantiated claims such as "agents are a better technology than objects". The team leader is a senior SEMAT member (i.e. one of the 31 individual Signatories) and the team members are identified from the 800+ supporters. The team leader (or possible he/she and a very small support group) is responsible for writing a report (like a book chapter) based on input from the research team. Inter-group consistency can then be evaluated at the top level by SEMAT officials. Since this only occurs *after* the basics in the theory/kernel have been identified and agreed, the very existence of this underpinning conceptual model should ensure good quality outcomes.

Thus, in summary,
1) Concepts and relationships are defined by use of a metamodel and an ontology
2) Quantification relies on well-researched metrics
3) Justification is provided in terms of empirical evidence.
Together, these provide the essence of software engineering.

References

1. Brinkkemper, S., 1996, Method engineering: engineering of information systems development methods and tools, *Information and Software Technology,* **38(4)**, 275-280
2. Calero, C., Ruiz, F. and Piattini, M. (eds.), 2006, *Ontologies for Software Engineering and Software Technology*, Springer
3. Fenton, N.E., 1994, Software measurement: a necessary scientific basis, *IEEE Trans. Software Eng.*, **20**, 199-206
4. Gonzalez-Perez, C. and Henderson-Sellers, B., 2006, An ontology for software development methodologies and endeavours, Chapter 4 in *Ontologies in Software Engineering and Software Technology,* Springer, 123-152
5. Gonzalez-Perez, C. and Henderson-Sellers, B., 2008, *Metamodelling for Software Engineering*, J. Wiley & Sons
6. Green, P. and Rosemann, M., 2005, *Business Systems Analysis with Ontologies*, Idea Group
7. Henderson-Sellers, B., 1994, Methodologies - frameworks for OO success, *American Programmer,* **7(10),** 2-11
8. Henderson-Sellers, B., 1996, *Object-Oriented Metrics. Measures of Complexity,* Prentice Hall.
9. Henderson-Sellers, B., 2003, Method engineering for OO system development, *Comm. ACM,* **46(10),** 73-78
10. Henderson-Sellers, B. and Gonzalez-Perez, C., 2006, On the ease of extending a powertype-based methodology metamodel, in *Meta-Modelling and Ontologies. WoMM 2006,* LNI Volume P-96, 11-25
11. Hug, C., Front, A., Rieu, D. and Henderson-Sellers, B., 2009, A method to build information systems engineering process metamodels, *J. Systems Software***, 82(10),** 1730-1742
12. ISO/IEC, 1995, ISO/IEC 12207. *Information Technology: Software Life Cycle Processes*, ISO/IEC, Geneva
13. ISO/IEC, 2004, ISO/IEC 15504-1, *Software Process Assessment - Part 1: Concepts and Vocabulary.* ISO/IEC, Geneva
14. ISO/IEC 2007, ISO/IEC 24744. *Software Engineering: Metamodel for Development Methodologies.* ISO/IEC, Geneva
15. Jeusfeld, M.A., Jarke, M. and Mylopoulos, J. (eds.), 2009, *Metamodeling for Method Engineering*, the MIT Press.
16. Kruchten, Ph., 2007, A conceptual model of software development, from *Software Project Management with OpenUP*
17. Kumar, K. and Welke, R.J., 1992, Methodology engineering: a proposal for situation-specific methodology construction. In *Challenges and Strategies for Research in Systems Development*, J. Wiley & Sons, 257-269.
18. Odell, J., 1994, Power types,. *Journal of Object-Oriented Programming,* **7**(2), 8-12.
19. OMG, 2006, Business Process Definition MetaModel (BPDM), OMG Document bmi/2006-11-03
20. OMG, 2009a, Architecture-Driven Modernization (ADM): Software Metrics Meta-Model (SMM). FTF - Beta 1, OMG document ptc/2009-03-03
21. OMG, 2009b, Ontology Definition Metamodel Version 1.0, OMG document formal/2009-05-01
22. OMG, 2009c, Organization structure metamodel (OSM), OMG document bmi/2009-08-02
23. Ralyté, J.; Brinkkemper, S. and Henderson-Sellers, B. (eds.), 2007, *Situational Method Engineering: Fundamentals and Experiences.* Springer
24. Roberts, F.S., 1979, *Measurement Theory with Applications to Decisionmaking, Utility, and the Social Sciences,* Encyclopedia of Mathematics and Its Applications, Addison-Wesley
25. Whitmire, S.A., 1997, *Object Oriented Design Measurement*, J. Wiley and Sons, Inc.
26. Zachman, J.A., 1987, A framework for information systems architecture, *IBM Systems Journal,* **26(3)**, 276-292
27. Zendler, A., 2001, A preliminary software engineering theory as investigated by published experiments, *Empirical Software Eng.,* **6,** 161-180
28. Zuse, H., 1991, *Software Complexity. Measures and Methods*, Walter de Gruyter.