The first Semat workshop: a blueprint for Semat

Ivar Jacobson, Bertrand Meyer, Richard Soley

The first Semat workshop was held in Zurich on March 17-18, 2010, with an extended day on March 19, 2010. The first day included presenting Semat goals by Ivar Jacobson, Bertrand Meyer, and Richard Soley ("the Troika"), position paper presentations by all participants, and summary and analysis of position statements by track leaders. The second day continued position papers' summary and analysis, set up goals and proposals for further action, and had group discussions of each tracks. The last day focused on the discussions of two main issues: the requirements of the Semat initiative and the definition of Software Engineering.

The workshop was attended by 28 people, including industry signatories representing four corporations. The extra day had 15 people attended.

This report highlights the event of the first Semat workshop, including its purpose, activities, results, and future venues.

1. The purpose

The purpose and scope of Semat initiative are described in the Vision Statement (http://www.semat.org/pub/Main/WebHome/SEMAT-vision.pdf). Semat involves three principal groups of stakeholders: industry (including developers and executives), academics and methodologists. Prior to the workshop the only agreement we all had signed up for was the Call for Action. A Vision Statement had been distributed but not been deeply discussed

The purpose of the meeting was to agree on how to address the Call for Action.

The Troika had provided a Vision Statement well in advance of the meeting. Starting with the Call for Action, agreed upon by the whole Semat community, the original Vision Statement suggested five tracks along with which the work would proceed for the next few months. A leader had been identified for each track. The tracks and their leaders are:

1) Definitions (Brian Henderson-Sellers)
2) Theory (Dines Bjorner assisted by Michael Goedicke)
3) Universals (Ian Spence)
4) Kernel Language (Jean Bézivin)
5) Assessments (Watts Humphrey assisted by Paul McMahon)

The Troika published a Call for Position Statements on www.semat.org. After a review of all the submissions, a subset of position papers was selected for presentation at the workshop.

Given that the participants come from so many different disciplines and with different background and each one has his or her own objectives, the intention was to allow this first meeting to be open with as few constraints as possible, starting the discussion from the individuals input. The goal was to offer participants a chance to present their position statements, and produce, as a result of the meeting, a separation of work into tracks, with a clear mandate to each track leader.

2. The Activities

The first day started with a presentation of the Semat goals by Richard Soley and continued with the presentation of position statements, 25 in all, by participants. All the position statements can be found at http://www.semat.org/bin/view/Main/WorkshopPosition.

The following sessions were each devoted to a track leader, starting with a summary by the track leader of the information in the relevant position statements and continuing with a full-group discussion on future directions of the track. These discussions were not expected to lead to any concrete results, but to give the whole group a sense of everyone's ideas and the direction of the track. These sessions accounted for a major part of the workshop, lasting until the morning of the second day. The track summaries are attached as appendices to this document.

On the second day afternoon, Bertrand Meyer and Ivar Jacobson presented the goals and proposals for further action. Bertrand Meyer presented "Software Science & Engineering for Semat: a Semi-Personal Perspective". He stated that the main focus of Semat should be professional programming for enterprise applications, rather than "casual" software development. Meyer also emphasized the importance of technology; much of the progress in software engineering over past decades has resulted from better tools and languages (as well as better hardware support). A decisive characteristic of software engineering is, beyond the superficial differences, the fundamental unity of the software process: throughout the software lifecycle, we face some of the same issues, and can benefit from some of the same conceptual and technical solutions. Meyer proposed a tentative set of one-year goals along each of the five tracks: definitions (50 key concepts); theory (identification of 3 to 5 relevant theories, justification of their relevance, and identification of areas where theories are needed); universals (20 universals, justification of their relevance); kernel language (basic design, demonstrator applications); assessment (5 assessment techniques).

Ivar Jacobson's talk was "Where we are going!" His talk focused on the kernel and kernel languages. He talked about the paradigm shift from "process" to "practices". All methods comprise a set of things that are always there, documented or not. These set of "things" constitute the "kernel", which is the foundation of methods and practices. Such a kernel will allow the industry to get control of all the methods they have. They will be able to find the best practices in their different development teams, harvest them and reuse them. They will be able to move people from one product team to another without having to relearn almost everything. The objective is not to reduce the number of methods, but instead allow every team to select their own method from a set of practices and to write new practices quickly once created. Other key stakeholders are the developers, the researcher, the teacher and the methodologist, all of them benefitting from the kernel and practices.

Stephen Mellor presented his view of the relationship among patterns, methods, practices, assessment, project context, and kernel languages.

Bertrand Meyer led a discussion on how to move forward working with the parallel tracks, which led in particular to defining the topics and schedule for the third day, and discussing the next opportunities for meetings (see section 4).

On the third meeting day (the extra day) several sessions were held.

Scott Ambler started by leading a session on “requirements of the Semat initiative” The goal was to identify stakeholders of Semat, their roles and usages of Semat. This brainstorming session helped clarify the goals of Semat, and the different stakeholders: industry representatives (developers, executives), the research community, teachers, and methodologists.

Larry Constantine took the lead of a session on the track structure. The proposed track structure is open for debate. Alternatives were presented. One of the main results was to add to the Semat initiative a Governance Model that will be responsible for setting up policies, bylaws, and intellectual properties. It was also suggested to add a “social organization” to deal with public relations, supporters out reaching and to better serve a broader community.

Tom Gilb gave a presentation on how to use his Planning approach to plan the Semat effort and to identify its final objectives (requirements). He talked about building infrastructure versus products and the three levels concerns and responsibilities, i.e., fundamental objectives, strategic objectives and means objectives. A comment was that any such planning needs to start from the result of the requirement session.

Finally, the group discussed the immediate need of a definition of the term “software engineering”. This work would be part of the Definitions track. But since it will most likely take quite some time (e.g., one to two years) to get the agreement on a “final” definition, we agreed to adopt a tentative definition. We also agreed that our “final” definition should stand on the ground of definition of engineering in general. Thus we need to liaise our definition with the engineering community.

3. The Results and Actions Items

The Zurich workshop was the first meeting of the Semat community. It served as the requirements elicitation, validation and verification phase in software development world. It produced few tangible and directly usable results, but its main benefits are elsewhere: the workshop opened up the dialogue in the Semat community and defined the expectations.

The main results are the thorough discussion of the Vision Statement. This document carries the broad agreement of the Semat community, but a number of elements were identified that require adaptation.

Concrete results include the following:

- The Vision Statement was accepted in all essentials. It will guide us in the work going forward.
- The five tracks remains the same for the next a couple of months. A new track has been added: ‘The Semat Requirements’ track.
- As a tentative definition of software engineering, we adopted Tom Gilb’s: Software engineering is “the discipline of making software systems deliver the required value to all stakeholders.”
- The track leaders must start to select members to the teams. These members may come from the entire community, not just from the signatories or the participants at the meeting.
- Each track leader selects his way of working. Infrastructure in some form to simplify collaboration will be provided by Carlo Furia.

4. The Next Venue

The next regular Semat meeting will take place in Washington DC, USA, on July 13-14. Groups that want to meet face-to-face before they report their work, can meet on the 12th (the ”extra” day).

In addition, working sessions gathering some of the participants will take place at various conferences:

- International Conference on Software Engineering (ICSE), in Cape Town, South Africa, where the organizers have invited us to organize a Semat workshop, to be led by Bertrand Meyer (Friday, May 7, http://www.sbs.co.za/ICSE2010/, see “Semat” entry).
- SEAFOOD (Software Engineering Advances For Outsourced and Offshore Development), Saint Petersburg, Russia, 17-18 June, with keynotes by Ivar Jacobson, Richard Soley and Bertrand Meyer (http://seafood.ethz.ch).
- TOOLS EUROPE, Málaga, June 29-July 3 (http://malaga2010.lcc.uma.es/), with keynotes to be confirmed.

Participants who will attend any of these events should contact the respective speakers to help organize the Semat sessions.

Appendix 1: Definitions

Prepared by: Brian Henderson-Sellers

The 90 minute session was designed to be a 10 minute presentation by the track leader and then open discussion on the track topic.

In the introduction, an example of the definition for "Team" was used to illustrate not only how a definition is partly relevant to the kernel language and partly to the kernel language usage (in the Method Domain) so that instances of that definition are relevant in the Endeavour Domain but also how important are the links to the other tracks, especially the kernel language and the universals tracks (Figure 1).. The notion of a definition was then linked to formal representations using Venn diagrams (set representations) and to powertype patterns that could, by agreement with the Kernel Language track, be the basic foundation of both the kernel language and the definitions.

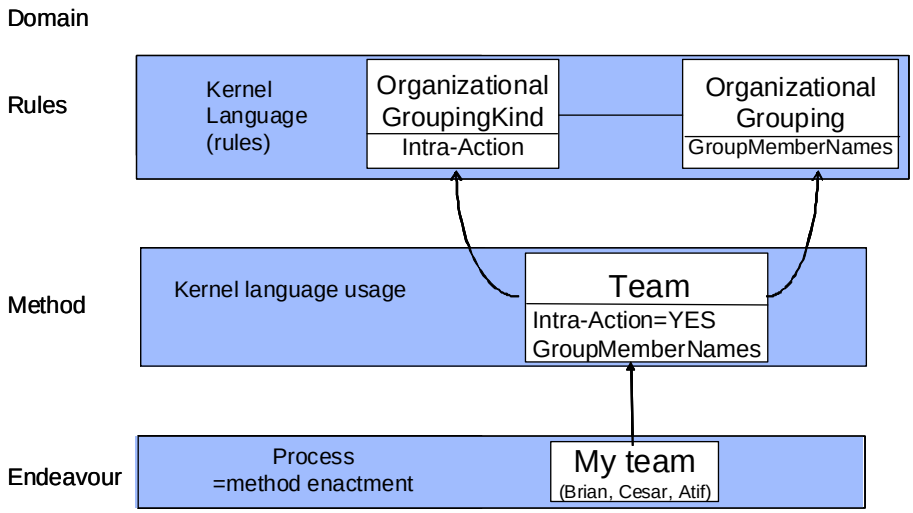


Figure 1 Relationship of definitions to kernel language and kernel language usage and to three domains of interest

Before the second part of the presentation (a survey of participants' position statements in the context of the Definitions Track), a general discussion ensued. Being the first track to be presented, the discussions were permitted to be far-ranging since the scope of definitions exceeds that of many of the other tracks. For example, some of the points raised were:

- Database interface seemed to be missing from all tracks
- User interface seemed to be missing from all tracks
- Perhaps the track structure is not optimal
- Is there too much overlap between tracks – should we reassess the track structure?[1]
- Should the "kernel" be ambiguous or not?
- We need something that works – using the interests and expertise of three companies represented at the meeting
- Should our efforts be top down or bottom up – or both?

[1] Later discussions on Friday resolved that the Track structure should remain and be enhanced by a SEMAT Requirements Track.

- How to allow for synonyms. Tom Gilb's suggestion here of a single definition with pointers to it from any synonyms seemed to be widely agreed.
- We need to avoid re-inventing things that already exist – but we need to identify our sources of definitions carefully.

The chair then presented his summary of what participants had said in their position statements relevant to the definitions track. He offered some long term objectives as well as ideas for discussion at the Zurich meeting.

The definition of software engineering is identified as highly contentious (this turned out to be a topic of the extended day on Friday at which meeting a tentative definition was accepted). Other common concerns or interests included some version of method engineering and method components; teams, team structures and other people issues; a need to avoid more "method wars"; the role of project management (whether in scope of SE or not); the vital need to include usability issues; the possible use of taxonomy, ontology and metamodelling as part of the formal representation of SE; and the persistent use of terms such as "release" in industry with multifarious meanings.

In the subsequent discussion, again far reaching, participants were concerned about their role and contribution to SEMAT; the need to agree on SEMAT objectives. (This was further discussed on the Friday session result in a matrix of users and usages of SEMAT initiatives) There was also a need to clarify the drive to derive a "kernel" and what that means. Should it be unambiguous? How is it related to the notion of kernel language and kernel language usage (as in Figure 1 and the Kernel Language Track). There is a need to discriminate carefully between the elements of the language (the definitions) and the application of those definitions e.g. in a method or enacted on a project.

The discussion concluded with no firm resolutions but plenty of ideas. Those wishing to participate in the work of this track should contact the Chair directly at brian.henderson-sellers@uts.edu.au

Appendix 2: Theory

Prepared by: Michael Goedicke for Dines Bjorner

First, the session chair presented a short list of properties regarding the kind of rigour / formalism one would like to have. Also the range and kind of theories should be discussed. It presented a wish list, which is not intended to name necessary conditions. On the list of desired properties are:

- rigor, math-based
- compositionality, interoperability and
- pragmatics of notations and methods

Important prerequisite

Important to note was the discussion about two topics:

1. range of the effort and where to start
2. degree of math-basedness of theories, the term *theory* in this context.

The discussion, which followed focused in a substantial part, whether the interests of SEMAT are solely on technology and hard math-based theories regarding the artefacts developed during a software development process.

As can be seen from the summary below and reported experiences from large projects it seems that we have more technical solutions around (which still need more research) than we have a good grip on the process / human related issues ("soft" aspects). This feeling can be substantiated by the fact that many of the reported project failures cannot be attributed to using wrong technology but to wrong or poor management decisions. This highlights the fact that an integrated view (technology ***and*** people) is needed and delineates the areas of research.

It was agreed by the group that software engineering is not only about these artefacts and products but encompasses also processes between human beings, organisation, management and collaboration to name just a few aspects of this category. Thus other scientific disciplines like sociology, psychology and management science need to be used and made instrumental to found our (SE) discipline in a proper way.

Summary in Detail

The following discussion can be summarized as follows:

1. There are already a number of useful theories in the field of Software Engineering (a list is given below). However, the discussion also revealed that these theories are often centred around the artefacts produced during a software development process. People and process related theories are necessary and useful but there is the need to develop an empirical basis for them.

A first list of existing theories was proposed during the meeting. It is tentative and most likely misses some relevant theories.

1. Theories of products
    - Programs
        - Construction
            - Software architecture (design patterns, Garlan/Shaw…)

        - Correctness-based (Dijkstra, Gries…)
    - Analysis
        - Static correctness (Hoare, model checking, …)
        - Dynamic correctness (testing, debugging)
        - Reliability (e.g. Musa)
    - Metrics (e.g. complexity)
  - Data
    - Relational
    - Abstract data types (algebraic, categorical)
  - Requirements
  - System behavior (performance analysis, e.g. networks)
2. Theories of processes
  - Lifecycle models (agile, CMMI)
  - Cost models (e.g. Cocomo)
  - Programmer behavior
  - User behavior

2. There was a broad discussion on the value of formalization and the use of mathematics in defining methods in Software Engineering. It became clear and the point was explicitly stressed that SEMAT was ***not*** meant to define only those methods and techniques within the range of SEMAT if there is a mathematical basis in the sense of mathematical logic or numerical functions. It was felt by the group that methods, guidelines and recommendations in Software Engineering can also be founded on an empirical basis e.g. using tools from psychology. This is especially useful and important for the emerging field of empirical software engineering. The current state of this field is already providing a good set of tools based in statistics delivering the necessary soundness and rigour for such methods, guidelines and recommendations. However, relevant experiments are expensive and sound data is not easy to come by.

3. There was also the discussion on whether this track is about an observing and explanation based ***science*** or a field of ***engineering,*** which is concerned with methods for building artefacts. While a number of primary concerns in Software Engineering are clearly engineering related there are also important aspects, which need a scientific explanation (e.g. in empirical Software Engineering?). This aspect was not discussed in adequate depth, however. The implicit feeling was that engineering aspects prevail, at least at the moment.

4. An interesting short discussion was about the role of theories at all. In (general, traditional) engineering theories deliver practical results in form of rules of thumbs. The author of these notes likes to add a quote (I saw this quote assigned to *James Clerk Maxwell* and to *Kurt Lewin, a* German-American psychologist): *there is nothing more practical than a good theory*.

5. The role of general Systems Theory was discussed to what extent it might be useful in the context of explaining artefacts and processes in software development.

Finally the discussion brought up what to do with these results.

- It was agreed to select the list above as a starter to draw a map of already existing theories regarding methods, tools and techniques in software engineering. It was regarded as useful to define attributes for the theories in order to classify them. This approach was chosen in contrast to the approach to develop a new set of theories from scratch. (Identify theories as islands in the ocean of software engineering)

- If possible, the links between theories should be identified (identify [possible] bridges between islands). This is to fulfil the wish that the theories should interoperate.
- Identify missing theories and missing links between existing theories.

One possible way to go forward with this is to establish an appropriately structured wiki. However, the discussion then showed that this should be discussed further later on.

Appendix 3: Universals

Prepared by: Ian Spence

Main Themes of the Position Papers

The main themes of the position papers can be summarized as:

1. Context is crucial
   – this includes the engineering, organizational and business context
2. A conceptual model of software engineering is required
   - and some have already been created
3. There are general engineering universals as well as software engineering ones
   - a relationship we can exploit
4. To be useful the universals must consider important indirectly related concerns such as people and project management
   - software engineering is more than just architecting and programming software solutions

Results of the Session

There are two main questions related to the universals track both of which were addressed during the session:

1. Are there any universal elements applicable to all software engineering endeavours?
2. What are the universals?

As there was overall consensus that there are a number of underlying universals applicable to all software engineering the rest of the session was spent brain-storming in teams in an attempt to build a bottom-up model of what software engineering entails.

Although working independently there was a lot of consistency between all of the results with elements related to (amongst others) being identified by all teams:

- People / Team
- Quality / Assessment
- System / Solution
- Plans / Lifecycle / Work
- Needs / Requirements
- Money / Resources
- Architecture / Design

As one would expect the vocabulary varied from team to team but there appears to be consensus about the type and extent of the things to be covered.

The original plan was to consolidate these initial results into a single model during the track break-out sessions but these were cancelled in favour of more plenary sessions.

The results will therefore need be written up and made available by the track lead.

Issues

1. The relevancy of the track – some attendees questioned the relevancy of the track feeling that it could be consumed into the other tracks. The consensus was that the track should continue as defined in the vision document
2. The relationship with the other tracks – there is obviously a close relationship with the other tracks which will need close co-ordination between the track leads
3. Providing different viewpoints – Pekka Abrahamsson notes a set of viewpoints that should be supported by SEMAT these include:
    a. Software as a knowledge machine
    b. Culture / cultural anthropology
    c. Engineering
    d. Scientific
    e. Mathematics (software can be expressed as an equation)
4. How do we share information – are knowledge sharing and communication tools going to be provided
5. How should results be presented – what format should the track results be presented in and who to

Getting involved in the track

So far the following people have signed-up to be involved in the Universals track:

1. Larry Constantine
2. Ed Seymour
3. Dave Cunningham
4. Miri Kajko-Mattsson
5. Shihong Huang
6. Jean-Marie Favre
7. Ivar Jacobson
8. Paul McMahon

Any other interested parties should contact ispence@ivarjacobson.com if they have time to help.

Appendix 4: Kernel Language

Prepared by: Jean Bezivin (INRIA) with the help of Jean-Marie Favre (OneTree Technologies)

Main Themes and position papers

It is very sad that Robin Milner, Turing-award and inventor of the Pi-calculus and type inference, passed away on Saturday 20 March 2010. Several SEMAT position papers insisted on the importance of languages in software engineering but none of them could be as concise and precise than Robin: "*Language is the raw material of software engineering, rather as water is the raw material for hydraulic engineering*". In June 2007, in the same building where the SEMAT meeting has been organized, Robin was presenting his Tower of Models Grand Challenge, focusing on the need to establish correspondences between these languages: "*These [models or languages], each involving a subset of the immense range of concepts needed for ubiquitous computer systems, should form the structure of our science. Even more importantly, the relationships (either formal or informal) among them are the cement that will hold our towers of models together*"

In the position papers, the importance of languages in software engineering was often mentioned. However these languages may take different forms. There are programming languages and modelling languages, visual languages and textual languages, general purpose languages and domain specific languages, object-oriented languages and functional languages, and many more categories. Language debates often turn out to be religious.

Typically software engineering was initially built around general purpose textual programming languages. Progressive abstraction allowed defining high level programming languages compiling to low level assembly languages, a first example of a precise correspondence between languages. Then the landscape was progressively populated by formal description languages, requirement languages, informal visual modelling languages, etc. Nowadays most software applications are written and maintained in many different languages, including several XML-based vocabularies or Human-Interface description languages. The problem of language composition is becoming a new issue.

Usually a general purpose textual language is used by a professional programmer (one that makes a living of programming). But the numbers of applications that will have to be built in the coming years for various purposes (individual, collective, familial, social, etc.) are in exponential grow. Many of these applications will have to be built by end-user programmers, using domain specific languages.

Results of the Session

A roundtable followed the track introductory presentation. Each SEMAT participant had about 1 minute to explain his/her position and ideas about the kernel language. Since similar comments occurred typically more than once, we provide below a consolidated list of topics instead of presenting the sequential list the participant statements. We indicate whenever possible the participants who contributed the different topics, but obviously since the arguments were sometime overlapping, the list of names below is just indicative. The summary below results from notes taken on the fly, so it might contains some misinterpretations and is certainly incomplete.

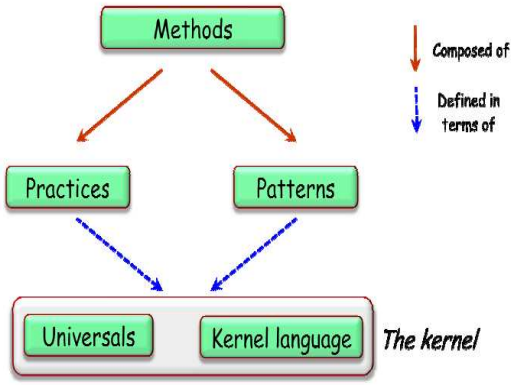
- **Natural Language** [MiraKajkoMattsson]. It may make some sense to start using natural language or a restricted version of it in the early version of SEMAT. This could help writing down practices and patterns. The problem with natural language is both its sequential forms

and the ambiguities, but this could be a first solution before the definition and consolidation of a better kernel language.

- **Meta Language**. [BrianHendersonSellers]. The Kernel Language should allow defining formally the concepts and their relationships.
- **Action Language**. [SteveMellor] The Kernel Language should be an action language, especially since the goal of SEMAT is to describe practices, processes, etc. Basic elements should include sets, state, transitions, but the syntax is not important. In fact, what is important is to come up with an abstract language and with the ability to define different languages on top of it if necessary. These languages should be connected not isolated.
- **Programming Language**. [BertrandMeyer] Programming Languages are more than just for programming. They can be used for modelling for instance. In that case, the program can be seen as a model. Because of huge experience in programming language design, so far, programming languages provide the best proven solution for Software Engineering.
- **Software Engineering vs. Programming** [PekkaAbrahamsson]. Software Engineering is not Computer Science and is not only about Programming.
- **Relation with standards** [JaanaNyfjord]. Standards are important in Software Industry. An idea might be to reuse a subset of the SPEM standard for instance.
- **Language Stakeholders** [IvarJacobson**]**. Various people with different skills are concerned by the kernel language and associated extensions or sub-languages. Stakeholders include: practitioners, developers, methodologists, computer scientists, language designers. Some are consumers. Others are readers, authors, etc.
- **Usability, scalability and language pragmatics.** [MichaelGoedicke]. Usability, scalability and language pragmatics should not be neglected. [BertrandMeyer] Programming Languages support scalability. [LarryConstantine] Usability and readability is a very important aspect in language design. This aspect should not be minimized. [PaulMcMahon] Languages should enable people to communicate. [IvarJacobson] The Kernel Language should be easy to use and appealing, not only formally defined.
- **Context of use and language variety. [**RobertPettit] Context is key. Kernel Language should be adapted to the context of use. [MarkKennaley] The kernel language architecture should make it possible to adapt the language(s) to suit the needs of different people, different clans, and different cultures. Defining jargons should be possible. [PaulMcMahon]
- **Language Community.** [IanSpence] It is very important that people can use the kernel language to contribute. The interface should be easy. People should be able to use their own vocabulary.
- **Linguistic Continuity. [**MeilirPageJones]. Languages to be used in the context of Software Engineering should go from natural languages to machine languages.
- **Linguistics = the Science of Languages.** [JeanMarieFavre] As pointed out by Robin Milner, *Language is the raw material of Software Engineering*. What is the Science of Languages? Linguistics. The Holistic View on Software Engineering should reuse when possible theories and results from Linguistics, not only from Computer Science.

Goals and roadmap

Based on all the previous considerations and the clarifying discussions at the first Zurich SEMAT meeting and since, we now have a clearer view of what are the goals and non-goals of the Kernel tracks. We can start defining a roadmap, in compatibility with the SEMAT Diamond:



1. The Kernel project as a whole has three goals implemented by the following tasks:
    a. To select and describe a metalanguage allowing to define and compose languages.
    b. To precisely identify the Universals (Universals track)
    c. To propose a language, based on the metalanguage, for defining and composing the Universals, ako generic software process kernel language with good extensibility properties.

    Tasks a. and c. will be performed in the Kernel Language track while task c. will be performed in the Universals track. Task c. will eventually need the results of both tasks a. and b. Work on all three tasks can start immediately.
2. The overall SEMAT goal is to create a platform (the kernel) allowing people to describe their current and future practices, patterns and methods so that they can be composed, simulated, applied, compared, evaluated, measured, taught and researched. Of particular interest to SEMAT is the generic software process definition language that may be used to define actors, stakeholders, methods, practices, patterns and universals discussed in the SEMAT vision statement (the so-called "process kernel language").
3. The global timescale for SEMAT is about 12 months. This means that the metalanguage and the Universals should be pretty sketched in the next three months in order to have time for definition of the process kernel language and further modification, evaluation and assessment through a number of use cases. We consider the process kernel language to be a big challenge for SEMAT.

Getting involved in the track

What can be said at this stage is that a number of attendees expressed interest in the definition of a kernel language for software engineering. As a consequence, the work is starting now and will be pursued by various means including face to face meetings for major milestones. The suggestion is to have our next track meeting at the TOOLS conference in Malaga. The results of this Malaga meeting could be reported at the main SEMAT common meeting.

Between now and the Malaga meeting, all those interested to contribute may contact Jean.Bezivin@inria.fr who is leading the metalanguage task. We are still looking for someone to lead the process kernel language definition in collaboration with Ian Spence of the Universals track. Some discussion channels will be established. Remember also that it is possible to contribute a position in the SEMAT blog moderated by the Troika (Ivar, Richard, Bertrand).

Appendix 5: Assessment

Prepared by Paul McMahon for Watts Humphrey

Main Themes of Position Paper's and what was discussed of importance to Assessment

There were two perspectives on assessment expressed:

a. Assessing SEMAT progress over next twelve months
b. Assessment features inside the SEMAT product

With respect to the first perspective following are the high points from the Track:

1. Need to measure progress over the next 12 months based on vision statement and priorities established.
2. Assessment track must work closely with all other tracks to monitor progress toward achieving the priorities established. (*See below under levels of agreement and suggestions from track for more details).*
3. Assessment track accepted the responsibility to coordinate a clarification of the objectives.
4. More specific measures will be determined as we move forward working closely with other tracks to help measure our overall progress.

With respect to the second perspective the following suggestions are an outcome of the Assessment Track held on Thursday:

1. It is critical that we measure progress with quantification and feedback.
2. SEMAT should explain why successful projects succeed and failing project fail and develop related measures to help.
3. We should identify where we need metrics based on what is useful and important to projects.
4. We should consider rules of thumb to help derive sensible measures.
5. Assessment should measure how we are doing at reducing the pain of end users.
6. SEMAT should provide a method to compare methodologies with the goals of quality, schedule, cost, satisfaction, and usability.
7. The Assessment group should discuss predictability further.
8. We should recognize that some measures will be temporary, and others will stand the test of time.
9. SEMAT Assessment should be based on:
    a. Improving software development and management
    b. Improving education on software development and management
    c. Improving consciousness in the community.

Levels of Agreement and Suggestions From Track

With respect to levels of agreement, and how the session went there were clearly different views expressed with respect to both the vision and the definition of software engineering. This only proves to show how important SEMAT is today.

Tracking progress over next twelve months will be based on priorities that still must be established.

The Assessment track needs to work closely with the new Requirements Track and the Universals Track (as well as other Tracks) to ensure we have an aligned vision and are assessing our progress accurately toward our goal.

People to Help With Assessment Track

With respect to inviting people to be members of the team and folks who are interested in participating in the ongoing working groups, I (Paul McMahon) have expressed an interest in continuing to help Watts in the Assessment track. Scott Ambler indicated that, while he is very busy, he would also like to stay in touch with the Assessment Track and possibly contribute to the extent he has time.

Appendix 6: List of Participants to the Workshop

This is the list of the 28 participants to the Zurich Workshop:

- Pekka Abrahamsson
- Scott Ambler
- Jorn Bettin
- Jean Bézivin
- Anders Caspar
- Alistair Cockburn
- Larry Constantine
- Dave Cuningham
- Jean-Marie Favre
- Carlo A. Furia
- Tom Gilb
- Michael Goedicke
- Brian Henderson-Sellers
- Carson Holmes
- Shihong Huang
- Ivar Jacobson
- Mira Kajko-Mattsson
- Mark Kennaley
- Paul McMahon
- Stephen Mellor
- Bertrand Meyer
- Martin Naedele
- Jaana Nyfjord
- Meilir Page-Jones
- Robert Pettit
- Ed Seymour
- Richard Soley
- Ian Spence