Supporting the SEMAT Initiative

Philipp W. Kutter (kutter@montages.com), David Zenie (zenie@montages.com), Michael Guttman (guttman@montages.com), Marc Moser (moser@montages.com).

SEMAT seeks to distill the core principles of software engineering into a flexible, extensible, and holistic conceptual framework that leverages commonly available standards. As long-time practitioners and theorists in this area, we at Montages broadly agree with SEMAT's vision statement. We also hope that we can contribute ideas, energy and perhaps even technology that will accelerate the progress of the SEMAT initiative.

Several years ago, we came to the conclusion - first individually and then collectively – that the chief challenge in software engineering is not information technology per se, but rather how to successfully manage all the work processes associated with the software engineering lifecycle. Since then, we have broadened that idea to include the work processes associated with the overall business engineering lifecycle, effectively making the field of software engineering just a specialized form of business engineering.

To be clear, what we mean by 'business engineering' is the set of processes by which businesses define and manage the operational processes that execute their line-of-business. For our purpose here, it is not the operational processes themselves that we are primarily interested in, but rather the business engineering processes that are executed in order to manage the definition and instantiation of those operational processes. In this sense, the various executing line-of-business operational processes are simply the ultimate work products of the business engineering lifecycle.

Obviously, such business engineering processes differ across different enterprises. For example, a pharmaceutical company defines and manages its operational line-of-business processes for manufacturing and marketing drugs in a different way than, say, an automobile company designs and manages its operational line-of-business processes for manufacturing and marketing automobiles. And, of course, even different companies in the same business domain will likely have different 'business engineering' processes, perhaps even markedly so.

As already noted, we see software engineering as just a specialized form of business engineering. However, it is important to note that part of that particular specialization is that the goal of software engineering is almost always to end up with executing software components that automate selected parts of the operational line-of-business processes of a given enterprise (or collection of enterprises). As such, a software engineering process designed to produce software to (for example) automate drug manufacturing must integrate well with the associated business engineering processes associated with designing and deploying drug manufacturing processes in general.

In this context, we look favorably on the SEMAT initiative as a way to begin formalizing and standardizing the 'language' – that is, concepts and terminology - of software engineering. As suggested above, we believe this will naturally lead to standardizing the 'language' of business engineering as well, which will ultimately allow software engineering to integrate more gracefully with business engineering. In short, what we eventually see emerging is a 'unified field theory' that can be used to formalize all the 'languages' associated with the many different specialized disciplines of business and software engineering, in such a way that it is also easy to specify and manage exactly how these different disciplines intersect and interrelate during their respective lifecycles.

In our opinion, emerging standards in the areas of formal modeling already provide a very good foundation for such a 'unified field theory' for formally specifying and then managing different but interrelated engineering disciplines. Using formal modeling standards like OMG's MOF and its various extensions, it is now possible to define many different 'domain specific languages' (DSLs), but manage all of them – and their interrelationships - in a consistent way.

To provide a very simplified example, suppose we begin by defining a DSL for the domain of business engineering in general. This DSL would then be specialized to define various more specific business engineering domains, including software engineering. Meanwhile, the common elements of these specialized DSLs would denote their intersecting relationships. Our own experiences with implementing this approach - using standard, open-source domain modeling tools - have so far proved very successful in allowing us to define and manage multiple intersecting engineering disciplines, and thereby improve the outcomes of the overall engineering process.

Of course, there may be other approaches to defining and integrating engineering domains that are equally valid. Our point here is that, on the one hand, an initiative like SEMAT is much needed to better formalize software engineering, and to begin bridging the current serious 'language gap' between business and software engineering, as well as among the many different flavors of software engineering itself. Meanwhile, on the other hand, the SEMAT initiative fortunately comes at a time where there are already widely accepted and standardized formalisms – with widely available supporting tools – that can readily be leveraged to help accomplish SEMAT's objectives.

In short, our own experience leads us to believe that SEMAT goals are not only important, but they are achievable. This makes us quite interested in supporting the SEMAT process.