The SEI Strategic Plan

featuring Bill Scherlis interviewed by Bill Pollak

Bill Pollak: Welcome to the SEI Podcast series, a production of the Carnegie Mellon Software Engineering Institute. The SEI is a federally funded research and development center at Carnegie Mellon University in Pittsburgh, Pennsylvania. A transcript of today's podcast is posted on the SEI website at www.sei.cmu.edu/podcasts. My name is Bill Pollak, and today I'm pleased to introduce you to Bill Scherlis, acting chief technology officer for the SEI. Bill also serves as director of an academic department within the Carnegie Mellon University School of Computer Science. In today's podcast, Bill will be giving us some highlights from the SEI's long-term strategic plan, which will be used to advance the practice of software engineering for the Department of Defense (DoD), federal agencies, industry, and academia through research and technology transition. Welcome, Bill; thanks for joining us today.

Bill Scherlis: Thanks Bill, glad to be here.

Bill Pollak: Please tell us, what are the biggest challenges faced by the SEI in the fields of software and cybersecurity in the next five years?

Bill Scherlis: Bill, our understanding of the challenges builds on several sources, obviously our own experience, but also reports to us from our many collaborators in the DoD, other government agencies, industry, and academia. It also builds on national studies from the Defense Science Board and the National Research Council, and most recently the study called Critical Code: Software Producibility for Defense.

On the basis of this understanding, we've identified three particular challenges that we're addressing in our strategy. The first one we call "architecture-led, incremental, iterative development" or "agile at scale." The point of this is to enable larger-scale iterative and incremental development, with acceptable levels of overall programmatic risk.

The idea is to be able to develop highly capable, rapidly evolving, innovative systems, but to do so in a way where the risk of completion of projects is within the bounds of acceptability for major systems developments.

This is facilitated through the early commitment to architectural structures. This is a key pattern that we have seen quite a bit, for example in the commercial sector, the idea of framework in apps, and we aspire to it in our world of defense software, for example in the framework of systems of systems.

So that's the first area. Actually, I want to say one more thing about that, which is that we often talk about software sustainability. When we think about the idea of continually evolving systems, we can think instead of sustaining a piece of software within a software-reliant system, instead we're supporting its continual evolution. So sustainability has a role in this area of architecture-led, incremental, iterative development. We actually, if you think about the acronym for that, it becomes ALIID, which reflects the kind of organizational structures that we would need to see in order to succeed in that dimension of the challenge.

The second area is "software assurance and certification," and this is a huge challenge for large-scale systems. There's a rule of thumb, which is that roughly half the cost of the software in a software-reliant system, in one way or another, has to do with quality assurance or verification and validation.

The current regimes for certification pose some challenges for the kinds of systems that we need to build now and in the future. One of the challenges is ongoing recertification, which is to say that once we have built a system, and we put it through what is now a very expensive and time-consuming process of evaluation and certification, we have certified just one snapshot of a system; but the reality is that for many of our systems, we need to support a continuous evolution. So we really need a regime of ongoing recertification.

There's another challenge in this area, which is the challenge that we call composability, which is "How can I take two separate components or two separate systems in a system of systems configuration, and certify them separately from each other, then put them together into a configuration, and then draw some conclusion about that overall configuration?" This is very important when we are linking systems together, in a sense creating kind of horizontal connections across systems, which is an increasingly significant goal for us. So this challenge of composability is also high on the list. Again, the current regimes that we have for assurance and certification really don't support that very effectively.

Another challenge in this area is the challenge of rich supply chains. It's hard to deny now that large-scale software systems draw on many sources of diverse geographies and origins from which we construct systems. We obtain components, or we obtain custom-developed systems,

subsystems. We get open-source software, and we use all of these pieces to create our systems. How can we develop an assurance regime that's going to support those kinds of rich systems and supply chains?

I just want to reference a Defense Science Board report that came out now five years ago, in fact exactly September of 2007, that was focused on this issue. There are a number of technical enablers to this, and we're beginning to see some real progress in this area. This is a hard problem, but it is not an impossible problem.

The third area of strategy is an area that we call "critical component capabilities," which is the development of particular kinds of capabilities that are critical for the defense mission, and to do so in a way that we achieve higher levels of capability, higher levels of predictability of outcomes, both in the process of development and in the execution of these particular kinds of components.

The most obvious one of these is cyber physical systems, what we used to call embedded. This is the recognition of the reality that much of the software that we build mediates between sensors and effectors, and there are diverse sensors and diverse effectors, and there are particular challenges in creating this kind of software, and particularly in scaling it up and achieving higher levels of assurance.

A secondary of critical component capability is support for autonomous systems, which is to say systems that can take that sensor data and go through a decision process, and actually enact outcomes without, or with some limited, human intervention.

In those cases, we face a tremendous challenge of assurance of verification and validation to ensure that these systems behave as we want them to behave, and that they do not behave as we do not want them to behave. This is a huge challenge because the state space, the number of possibilities of combinations of inputs and environmental circumstances for these systems is very high.

There's a third area of critical component capabilities, which is data intensive, highly distributed systems. High-performance computing and data-center cluster architectures are beginning to converge. In a certain sense that's very good news, but the software challenges, because of the high level of concurrency, the spreading out of data and computation, the reliance on communications networks, all of these make this a particular challenge, but it's also an area of particular importance to the DoD mission.

Bill Pollak: Okay thanks for that, that's great. So how does the SEI plan to address these challenges? And if you could give us some highlights of future areas of work that are pertinent.

Bill Scherlis: So we address these challenges from a management perspective, primarily by adapting our work effort to promote the gathering of data and experience that relates to these challenges, and also by promoting interactions across our traditional program areas.

So for example in the first area of strategy—architecture-led, incremental, iterative development—we have two of our programs—one is the RTSS Program, which has focused traditionally on architecture; and we have another program, the SEPM Program, which has focused traditionally on process and measurement—to succeed in addressing this challenge of architecture-led IID.

We really need to have strong interactions across these programs, but in fact we also need to bring in work from the NSS or CERT area on cyber security, because the architectural decisions strongly influence quality attribute outcomes and particularly security outcomes, so we want to make sure that as we are thinking about architecture in the early stages of the conceptualization of systems that we are addressing security issues.

And finally, with respect to the ASP Program focused on acquisition practices, that also needs to play because we need to understand the realities of the engagement between the government and its contractors. So that's one example of the essential requirement to promote these kinds of interactions across programs. Analogously we also need to promote interactions across the full spectrum of RDT&E from basic research, what we call 6.1 research, which is the research typically done by professors and graduate students in universities, all the way across to prime contractors leading development projects and operators who are responsible for the operation and sustainment of systems.

And finally, I want to point out a shift in the environment of software engineering research and development, which is that we are now a much more data-intensive discipline. If you look at the development environments that are now in common use in commercial practice, and even in the high-end open-source practice, you'll see that there's an enormous accumulation of data and support for diverse analytics on that data.

This is data that pertains to the development and operation of software systems. It's logging data and operations, it's also development data; for example, the history of every single line of code in multi-million-line-of-code systems is there in the database to be analyzed.

So we're now much more involved in this kind of data analysis to help us understand where we can be more effective in developing interventions. So those are three examples of how we're addressing the challenges.

Bill Pollak: Okay great. Does the strategic plan include metrics to ensure that the SEI's research focuses on significant and pervasive problems for the DoD? And if so, could you give us a few examples of how those metrics will be applied?

Bill Scherlis: Yes, that's a very good question; it's a very important question. Software, as we all know, is an area where the measurement challenge is profound. Everybody has heard "the light under the lamppost" story. We have many lampposts, but where they shed light is often not very useful, and we have many areas of darkness where we hope to build lampposts where we can shed light that will be useful.

So in our internal evaluation process for example, for new research projects, we apply a number of criteria. For example, we have four validity criteria that we apply. Two kinds of operational validity, one that we call *operational scope*: "Are we actually working on the right problem? Have we picked a problem that matters to somebody who's trying to accomplish the mission?" Two is *field significance* which is "Were we to succeed in this research, do we have a way to assess whether our solution will actually have the intended impact in the field?"

So both of those dimensions of validity relate to operational characteristics. But we also want to make sure that what we're doing has the right technical characteristics, and particularly *technical soundness*, which is "Is the technical approach scientifically founded? That is, are the conclusions that we're reaching sound and actionable conclusions from a technical standpoint?"

And then finally, *technical significance*, which is "Are we doing something that is actually well situated in the discipline, that is recognizing of the previous work in the technical literature where we have the right set of partners?" So the work we're doing actually is a step forward, and we're not repeating work that has already been done elsewhere.

So those are four validity criteria, but of course we also think in terms of the very familiar Heilmeier questions. These are from George Heilmeier who was a former DARPA director, and he came up with a very short pithy list of questions to ask that research managers can use to help make sure that when we're developing plans of research that we're focusing on problems that matter, and that we are engaging in the research with acceptable levels of risk and observability.

So, for example, one of the key Heilmeier questions is "What is the mid-term exam? How do we know partway through a project whether we're getting close to a solution or not?" We can't simply invest for several years and hope for the best.

So we have to be attentive to how we manage. On the other hand, we don't want to over-manage, because we need to let new ideas emerge; and we need to promote the flow of invention, innovation, and creative thinking, which is really essential, particularly in the software discipline where the pace of innovation really is unchecked. There is no plateau anytime soon in the field of software.

So we really have to make sure that in our management approach we recognize that. We don't think that we're getting to a point of diminishing returns because we're not; there's so much ahead of us in this area of software.

Bill Pollak: Okay, Bill, could you give us some highlights of the exploratory research the technologists are conducting at the SEI and how it's guiding future areas of research?

Bill Scherlis: Yes I'll just—I'll briefly highlight three examples of projects going on here.

Bill Pollak: Great.

Bill Scherlis: Just to give a sense of what we do.

So one project is focused on adapting cloud-based architectures: these vastly distributed cluster configurations where there are many, many disk drives and many, many processors and the data is spread out. Adapting those ideas which we normally think of as associated with giant data centers, situated near power stations, because they're so enormous.

Well, how do we adapt those ideas to the tactical environment where we have forward-deployed configurations of computing and storage and communications, perhaps intermittent connectivity for reach-back support? So how can we think in terms of forward deployment of these capabilities in the tactical setting? That's one example.

A second example is advanced software analysis, which is moving beyond the old-fashioned ways of simple testing and inspection to assess the quality of code. We need those very much, but we need to augment them with new capabilities that are focused a little bit more on the technical, mathematical dimensions of software, where we can do direct analysis of code or where we can monitor execution in a very rich way through dynamic analysis.

And the interesting thing about this set of technologies is that we use them, on the one hand, to support software assurance for the systems that we're developing, at code level. But, on the other hand, we use those same techniques to support the analysis of malware, which we have a large collection of, to identify common features, and ultimately try to map out the supply chain.

You know software is the *material* for cyber defense and cyber offense both, so cyber security is a really essential part of the mission here. When I spoke earlier, for example about software assurance and certification, that's an activity that's going on in every program in the SEI. For example, in the NSS Program, there's a project that is focused on developing secure coding standards for the C and C++ languages, and they're now starting with Java. And what's interesting about that is that they achieve impact not only through identifying best practices, but they're also directly engaged in the standard setting project for these languages, C language in particular, so they can influence the later users of one of the most pervasively adopted programming languages that's out there, in a way that promotes the development of secure

systems out of the box. That's kind of the goal. Anyway, that's the second area is software analysis.

The third area that I want to mention relates to data that's associated with the software development process, and the particular example I want to cite here has to do with cost data, which is data that is developed both by those who are proposing to develop a system and also those who are evaluating proposals.

And this is very rich data that tries to parse out from a set of requirements and goals for a system, some sense of how much it might cost to build it, and also how much risk or uncertainty there is associated with that. Another way of thinking that is, What's the variance on the cost estimate? Is there a wide variance or a narrow variance?

Well traditionally, that's a process that's done at the very beginning of a project and then we kind of set it aside and we push on as we do our development. But in fact if we're doing this iterative incremental development—as we had talked about earlier, the architecture-led—what we really want to do is have a kind of continuous process of re-estimating costs. Every time we make a design commitment, we build prototypes, we resolve uncertainties, we do experiments, we resolve more uncertainties, then those activities to reduce engineering risks in a sense narrows out that distribution curve, and then we've also incurred some costs so we want to think about what's the cost to complete the project. So we want to think about this idea of continuous re-estimation of both cost and risk in an iterative development process.

So those are the three examples of projects; and each of those involves interactions across the traditional SEI programs. Each of those also involves considerable collaboration and engagement across the spectrum, from folks doing sort of basic science research through to folks involved directly with the mission who can help us ensure that we're getting the right kinds of operational validity.

Bill Pollak: Well, Bill, thanks for joining us today.

If you'd like more information about the SEI's recent research, you can download technical reports and notes at www.sei.cmu.edu/library/reportspapers.cfm. This podcast is also available on the SEI website at www.sei.cmu.edu/podcasts and on Carnegie Mellon University's iTunes U site.

As always if you have any questions, please don't hesitate to e-mail us at info@sei.cmu.edu. Thank you.